THE DEVOPS ENGINEER'S CAREER GUIDE

A Handbook for Entry- Level Professionals to get into Continuous Delivery Roles for Agile Software Development

CAREER

DEVOPS

TECHNOLOGY

APPLICATION DEVELOPMENT

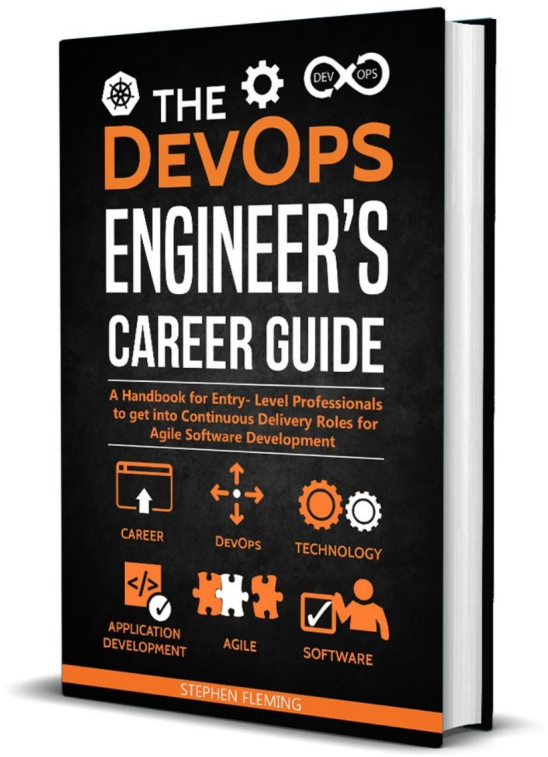
AGILE

SOFTWARE

STEPHEN FLEMING

The DevOps Engineer's Career Guide

A Handbook for Entry- Level Professionals to get into Continuous Delivery Roles for Agile Software Development



© Copyright 2019 - All rights reserved.

The content contained within this book may not be reproduced, duplicated or transmitted without direct written permission from the author or the publisher.

Under no circumstances will any blame or legal responsibility be held against the publisher, or author, for any damages, reparation, or monetary loss due to the information contained within this book. Either directly or indirectly.

Legal Notice:

This book is copyright protected. This book is only for personal use. You cannot amend, distribute, sell, use, quote or paraphrase any part, or the content within this book, without the consent of the author or publisher.

Disclaimer Notice:

Please note the information contained within this document is for educational and entertainment purposes only. All effort has been executed to present accurate, up to date, and reliable, complete information. No warranties of any kind are declared or implied. Readers acknowledge that the author is not engaging in the rendering of legal, financial, medical or professional advice. The content within this book has been derived from various sources. Please consult a licensed professional before attempting any techniques outlined in this book.

By reading this document, the reader agrees that under no circumstances is the author responsible for any losses, direct or indirect, which are incurred as a result of the use of information contained within this document, including, but not limited to, — errors, omissions, or inaccuracies.

Table of Contents

Table of Contents
Welcome
Introduction
Chapter One: An Introduction to DevOps
Chapter Two: Facts and Benefits of DevOps
Chapter Three: An Introduction to Agile Management
Chapter Four: DevOps Principles and Concepts
Chapter Five: Essential Skills for the DevOps Engineer
Chapter Six: Different Types Of Deployment
Chapter Seven: DevOps Engineer versus Other Engineers
Chapter Eight: Micro-Service and DevOps Engineers
Chapter Nine: DevOps Tools
Chapter Ten: The Career Path
Chapter Eleven: How To Market Yourself As A DevOps Engineer?
Chapter Twelve: Interview Questions
Conclusion
Sources

Welcome

Dear Friend,

It’s a privilege to assist you on your journey towards continuous improvement.

I am providing you link to the additional ***Technology Booklet*** which contains latest technology updates on Technology and Application Development Methodology. Topics Covered: Blockchain, DevOps, Microservices, and SRE Skills & Jobs.

“Get Instant Access to Free Booklet and Future Updates”

“Click Here”

List of my Other Books

Technology

- DevOps Handbook
- Microservices Architecture Handbook
- Blockchain Technology
- DevOps and Microservices
- Blockchain with DevOps & Microservices
- Kubernetes Handbook
- SRE Handbook

Mind and body

- Love Yourself
- Intermittent Fasting

Also if You Prefer the Audible version below are the available books:

1. Site Reliability Engineering (SRE) Handbook
2. Microservices with Kubernetes (2 Manuscript bundle)
3. Microservices Architecture Handbook
4. Kubernetes Handbook
5. Blockchain & DevOps (2 Manuscript Bundle)
6. DevOps Handbook
7. Blockchain Technology
8. Love Yourself
9. Intermittent Fasting

##You can check all my e- Books, Paperback and Audible at my **author page:** Click Here

Facebook: **@sflemingauthor**

Email: ***valueadd2life@gmail.com.***

Introduction

Hello! How are you and how is your Continuous Delivery journey going on? Are there any new skills that you want to acquire this year? If you do, please let me know what they are on my email.

My earlier books were on the following topics: DevOps, Microservices, and Kubernetes & Site Reliability Engineering.

In the last four months, I have been heavily involved in the recruitment process of various DevOps related jobs in my current project. I have come across multiple Entry Level and Mid-Level career professionals inquisitive about expectations of the role and how their earlier experience would contribute to the DevOps role. Also, I have received several emails from readers asking how to switch from their existing roles (development, sys admin, etc.). Based on the interactions, I have included “DevOps Engineer” related queries in the below categories and in this book, I will give you complete information about the position, career path and skill set required.

The main queries were the following:

- *Why DevOps?*
- *What are the job duties and day-to-day activities of a DevOps Engineer?*
- *What did DevOps engineers do before DevOps?*
- *What technical and soft skills are required to be an expert-level DevOps Engineer?*
- *What are some standard tools a DevOps engineer uses?*
- *What are other similar roles from where one can make the transition to the DevOps world?*
- *What are the Certifications/Courses one can do to become a DevOps Engineer?*
- *How can I get DevOps interviews with top companies?*
- *What are the average Salary, companies to work for, and*

designations/roles?

- *How is the career path of a "DevOps Engineer"? How is the career advancement of a DevOps engineer?*

The book covers most of this information. Over the course of the book, you will gather information on what DevOps is, and how you can use it to improve your processes. You will also identify the different roles that are linked to DevOps. If you are keen on becoming a DevOps engineer, the last few chapters include information on what skills you need to develop and what path you need to choose. Also, the last chapter contains sample interview questions, which are the most common ones asked during a DevOps interview.

Overall, this book is aimed at professionals (0-5 years of experience) looking for DevOps role overview in limited timeframe. If you have to connect the dots regarding your existing experience, credentials and its fitment/relationship with the DevOps role, it would provide you much needed clarity. It also talks about other similar and related roles and its relationship with DevOps role. Also, if you are part of Project Management Team or Business Development Team or recruitment team (HR) this book will provide you required information about the DevOps role.

Thank you for purchasing the book. I hope that you get all the information you are looking for.

Chapter One: An Introduction to DevOps

DevOps is a collection of numerous concepts that are not entirely new. The concepts and techniques have created a change in the technical community and have spread across numerous projects. Since this term is not very popular, many people have confused its meaning and have the wrong impression of DevOps. This chapter will define DevOps by discussing its standard framework and the numerous areas that it covers. Like “Agile” and “Quality,” DevOps is a concept that takes some time for an amateur to grasp. We will cover the basics of Agile in the third chapter of the book.

Definition of DevOps

A new abbreviation or term called DevOps was developed with the blending of two significant trends. The first of these trends is called agile infrastructure or agile operations, which sprung from the application of the Agile and Lean approaches to projects. The second trend is a deeper understanding of the collaboration between the development and operations staff when creating or operating a service, through every stage of the process. DevOps also takes into account how essential operations have become for any business, as every industry is now customer-centric and service-oriented.

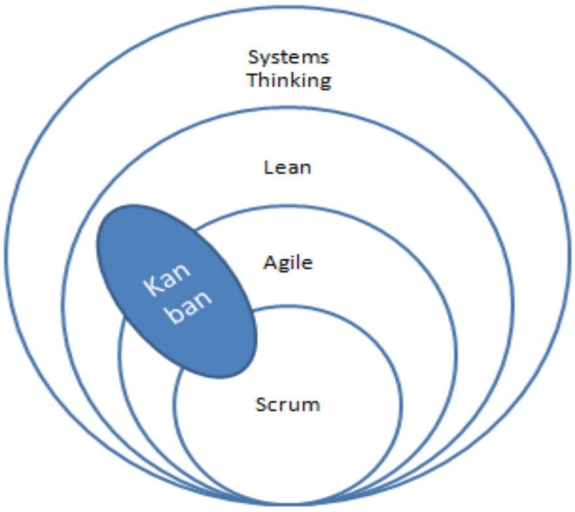
Jez Humble defined DevOps as a practice that uses information from different disciplines to study systems. The concept of DevOps is the study of the development and operation of rapidly changing systems. This is a good definition of DevOps, but for a startup or small company, it is too complicated. It would be best to define DevOps as a practice of development and operations engineers working together during the lifecycle of a process, including the development of the product and customer support. DevOps also has some techniques that both the operations and development staff use to make the systems work. These techniques range from participating in Agile development processes to using tools to test the process.

That’s why there is no differentiation between different sub-disciplines in administration. The term “Ops” includes system administrators, system engineers, operations staff, release engineers, network engineers, DBAs, security professionals, ethical hackers and other job titles and sub-disciplines. The term

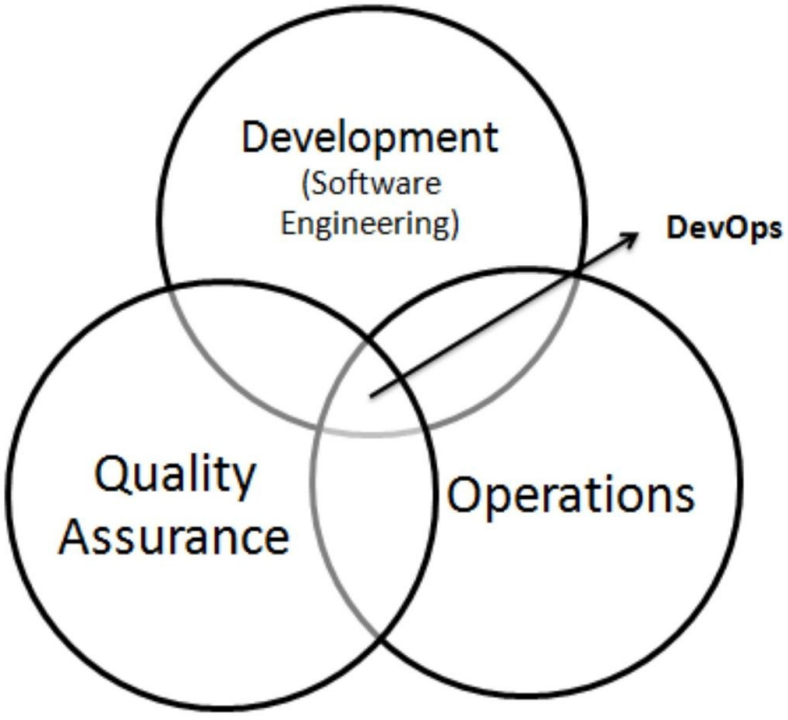
“Dev” is an abbreviation of the term Developers, but it means so much more than that. The term “Dev” is a blanket term for all the employees or staff who are involved in the development of a product. These employees include Quality Analysts, Product Analysts and other employees from different disciplines.

DevOps is closely related to the Lean and Agile approaches. Earlier, the operations always focused on improving productivity and the end product. Not many businesses realized that this approach does not benefit the company and leads to further disarray within the firm. To overcome this, DevOps was created. You can look at DevOps as a branch of the Agile software. The Agile software development tool looks at how the product management team, developers, quality analysts, and customers collaborate to develop a better product. DevOps agrees that this collaboration is necessary, but it also states that the firm should provide the clients or customers with information about the service delivery and how the system and applications interact with one another. This means that the product management team must make this a high priority. If you look at it from this perspective, DevOps refers to extending the principles of Agile beyond the boundaries of simply setting the application up to the entire service the team needs to deliver.

Since the discussion around DevOps covers a lot of ground, people define DevOps in many different ways, since it means different things to them. Some people say that DevOps allows the team to treat the code as infrastructure, improves the collaboration between the operations team and development team, automates processes, improves culture, or is simply an alternative to Kanban. The only way one can define DevOps is by defining the term agile development. According to Wikipedia and the agile manifesto, Agile development includes four important factors. To define DevOps, I have included a fifth level called the tooling level. Most people do not like to talk about tools within the Agile and DevOps framework, but it is also difficult to ignore them.



Agile Scrum & Kanban



DevOps Definition

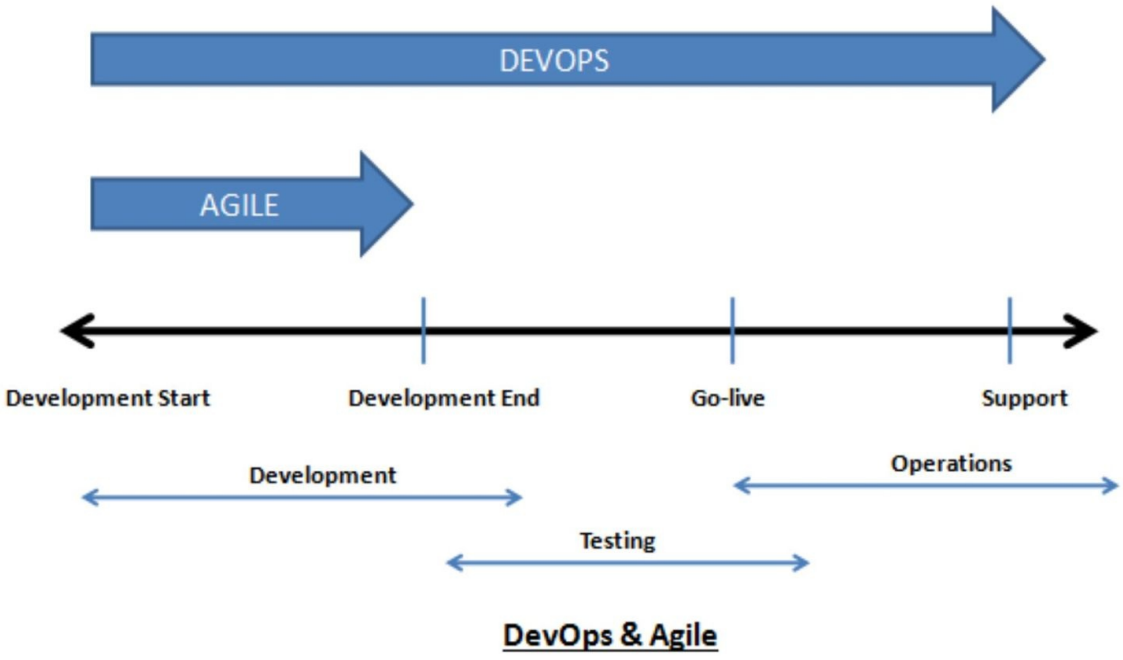
Agile Values

Agile values, which are included in the Agile Manifesto, are core values that

have to be agreed upon. In the following chapter, we will shed some more light on these values.

Agile Principles

Agile Principles are the strategic approaches that support agile values. The Agile Manifesto cites many specific principles which are listed in the next chapter. You do not have to use all these principles, but if you do not abide by some of them, you will not be applying Agile to your process.



Agile Methods

There are more specific processes to implement agile principles, which can be done with tools like Scrum and XP. Remember, none of these methods are mandatory, they're just possible ways to implement these principles.

Agile Practices

Agile practices are highly specific tactical techniques that are used along with Agile implementations. These practices do not have to be Agile, but by adopting these practices, you can increase the value of Agile implementations. These include planning poker, stand-ups, backlog and other artifacts that a developer will use to perform their work.

Agile Tools

In order to improve their work, some teams use tools like planningpoker.com, Greenhopper and other Agile tools to implement Agile practices.

Agile uses a top-down approach where the higher levels inform the lower levels. If an organization or a team pick up tools and practices without fully understanding the fundamentals, they cannot reap the benefits. This is because, in the past, this approach has never worked. People always talk about DevOps in parts, but all the parts will map to the principles and values of DevOps.

DevOps Values

The fundamental values of DevOps can be found in the Agile Manifesto. However, instead of only making it a working software, there is a slight change to focus on the overall software or service that is delivered to the customer. Alex Honor defined DevOps as ‘People over Processes over Tools.’ The DevOps values echo the Agile values and urge the collaboration between the development and operations team.

DevOps Principles

There is no single list of principles, but some of the widely accepted ones are covered in Chapter Three. One of the universal principles of DevOps is “infrastructure as code.” Experts believe that DevOps is a branch of the Agile principles since it also includes the collaboration between systems and operations, and does not only stop at checking the code.

DevOps Methods

You can use Kanban, Scrum, and other Agile methods for operations. The focus is on integrating the operations with the development, quality analyst and product teams. There are also other distinct methods like Incident Command Systems and Visible Ops. The former works towards changing control within the team while the latter works on recording any incidents. A combination of these methodologies is growing, and experts are trying to develop a more thoughtful process to monitor them.

DevOps Practices

There are specific techniques that you can use to implement the above processes and concepts. Configuration management, continuous integrations, continuous deployment, toolchain approach to tooling, monitoring and metrics schemes and cloud computing and visualization are some of the standard practices that one uses to bring change using DevOps.

DevOps Tools

The DevOps Tools are tools you will use to commission the principles. Recently, many tools have been released in the DevOps world. The following are some of the ones developed for different departments:

- To monitor releases – Jenkins, TeamCity, Travis
- For configuration management – Chef, Puppet, Cfengine, Ansible
- For orchestration – Noah, Mesos, Zookeeper
- For containerization, monitoring and visualization – AWS, Vagrant, Docker, and OpenStack

It is incorrect to call any tool a DevOps tool as it will not magically help you follow the principles and values of DevOps. Some specific tools are being developed to facilitate the implementation of the principles, practices, and methods of DevOps thereby helping you develop a holistic understanding of DevOps.

It is difficult to define DevOps, just like it is difficult to define Agile, but it is worth doing. When you look at the definition on a philosophical level, it can seem like an empty statement. Some people may wonder why they are being asked to do their jobs better. You must remember that just because you follow some instructions in a book, it does not mean that you are developing an Agile process. If you want to be a DevOps Engineer or an Agile practitioner, you must understand all the layers that go into making you an engineer, and how you can implement these tools and practices to help you develop a robust process. What DevOps wants to bring into Agile is the notion that the product or software is not fully developed until it meets the expectations of the users or customers.

When people talk about DevOps, three primary areas of practice that are often discussed are the following:

Infrastructure Automation

Here, you will need to create the systems, configure the operating system and use different application deployments as your code.

Continuous Delivery

In this practice, you need to build, test and deploy the application in an automated and speedy way.

Site Reliability Engineer

The site reliability engineer will operate the systems, monitor how they function and design the systems to allow collaboration between different teams.

History of DevOps

DevOps was developed when people and firms realized that they needed to innovate and automate processes on the technology side. The DevOps movement only started after the Enterprise Systems Management (ESM) and System Administration movement.

ESM rose in the 2000s, and it made people wonder what they could do to improve the processes. Companies and firms realized that their systems were still in the primitive state despite the effort that the employees put into improving the systems. Then, they began to look for different ways to improve these systems. John Willis, Mark Hinkle and Whurley from Zenoss sponsored a Barcamp around that concept.

In the year 2008, O'Reilly held the first Velocity conference which focused on the performance and operations of the internet. It allowed professionals from different fields to share information about the best practices for operations. In the year 2009, there were presentations about the advantages of a collaboration between the development and operations team in large businesses. They also talked about how this collaboration promoted a rapid and safe change in web environments. Tools like Chef and Puppet were marketed the most, and people then began to wonder how they could use these new concepts and implement them in their organization.

In parallel, in the development space, Agile development processes began to grow in popularity. As most organizations were talking about Agile Systems Administration, Agile processes moved from being a niche practice to a more

common one. This was especially true in Europe. Gordon Banner had addressed this in the early 2000s during a meet. Most of the focus of this movement was on the processes and the analogies from lean manufacturing processes and Kanban. In the year 2009, Patrick Debois and Andrew Clay Shafer sat down and talked about the collaboration between the development and operations teams and coined the term DevOps. Then, Patrick held a DevOps Days event in Ghent which lit the fuse. Now that the concept had a name, many people began to talk about it, and it was discussed in many events including Velocity.

In Patrick Debois' view, the concept of DevOps rose like a reaction against the inflexibilities and silos due to existing practices. This does sound familiar. To understand DevOps better, you should read more articles on the internet about the history of the movement.

Chapter Two: Facts and Benefits of DevOps

Benefits of DevOps

Now that you have an idea what DevOps is, let’s look at the benefits that a business can gain by deploying DevOps.

Quicker Deployment of New Applications and Systems

If you have successfully incorporated DevOps into your business, you should now work towards the next level of deployment. If you use the right approaches, you can ensure that your organization deploys its new systems in an enhanced and efficient manner. This can be done while maintaining and even sometimes maximizing efficiency. This is how continuous deployment and innovations become synonymous with each other. This makes it easier to deploy products and services faster.

Agility Is Key

It is essential for the business to be agile for it to go through a transformation. Through DevOps, any organization, regardless of its size, can transform the business into a more Agile system, which will allow it to achieve scalability. In simpler words, everybody wins.

DevOps Directly Translates Into Money

Through DevOps, a business can automate repetitive tasks without worrying too much about any errors. For instance, performance and regression testing can quickly bring about small changes. Frequent rollovers and backups in the development chain will lead to a more stable and robust process. When you have such automation in place, the organization can benefit by saving a large chunk of its manual costs. This therefore directly translates to saving money.

With DevOps, Silos Can No Longer Exist

Silos and innovation are things of the past. In today’s world of business, there is more to innovation than meets the eye. In the past, there was never a link between the operations and development team, which meant that any innovation was carried out in complete secrecy. This made it difficult for an organization to maintain transparency. Times have changed, and so have the methods of

innovation processes. Since there is an increased level of interaction between different teams in the organization, there is now more exposure. This exposure enables transparency between teams which creates a useful collaboration. This change can be brought about by following Agile methodologies since they have given a new meaning to innovation, thereby making it easier for employees. In a typical IT organization, there are many barriers that one must break with the right approach. Through the introduction of DevOps, an organization can get rid of old linear processes and introduce new processes that will improve the functions of the company.

Faster Development Cycles for the Organization

Communication and collaboration are the two main keywords relating to DevOps. In an organization, when these features are enhanced, there is an instant improvement in development cycles. This helps the organization achieve success regardless of what the project may be.

Continuous Service Delivery

Through DevOps, the organization can ensure shorter development cycles. This means that after they go through a series of tests, the codes are quickly released to the end users. There is very little or no gap between gathering requirements and the production cycle, which helps to create a massive shift in a production cycle. Using these methods, the organization can synchronize IT mechanisms with production cycles, thereby streamlining the cycle. This helps to improve the efficiency of the process, and that’s why an effective DevOps method should be incorporated into the system to create a stable and robust method to check efficiency.

Adieu Defects

In every application production environment, defects are your worst enemy. When you incorporate DevOps methods into your processes, you can reduce the number of defects to zero, and that’s why most organizations are now using DevOps methods to develop software. Through iterative development, modular programming and collaboration, teams can minimize any defects, thereby developing a foolproof method to handle or overcome defects. There are many alternative methods that the organization can use to minimize defects, which helps to increase efficiency in every aspect of the organization.

With so many benefits to using DevOps methods, people now believe that

DevOps is the future of the production cycle. If it is implemented correctly, organizations can achieve a lot in a short period.

What is DevOps Not?

It's Not NoOps

Some people believe that DevOps means that the developer takes over the operations tasks. They worry that the developers are taking over their jobs. Not completely, but there is some truth to that.

It is a misconception that DevOps is a working method that begins from the development side that is used to wipe out operations. You must remember that DevOps, like Agile and other processes, is initiated by the operations team. This is because the operations team realized that the existing processes, principles, and practices do not help in developing successful projects. Since development teams and businesses need more agility, the business climate becomes more fast-paced. Since the climate is rapidly changing, we provide less to the end user since we are trying our best to solve problems with more rigidity. This means that we need a fundamental orientation to provide the teams with the necessary infrastructure.

It is only now that we realize that there are some parts of the operations process that we need to automate. This means that the operations team can work on automating processes, or it can be done by the developers by writing code to build new ones. There are times when both the developers and operations work together to build robust processes. This may seem scary to some people, but it is a part of the collaborative approach. Teams will be successful only when the development team and the operations team work together to create an overall product.

It's Not (Just) Tools

DevOps is not just the implementation of tools. It is essential to define DevOps since many people misunderstand the term, using a poorly structured definition that ignores the theory and implements the many tools and processes of DevOps without keeping the principles in mind. This is an antipattern. You must remember that automation is just one way to exercise power, and if you do not automate processes wisely, you will damage the system.

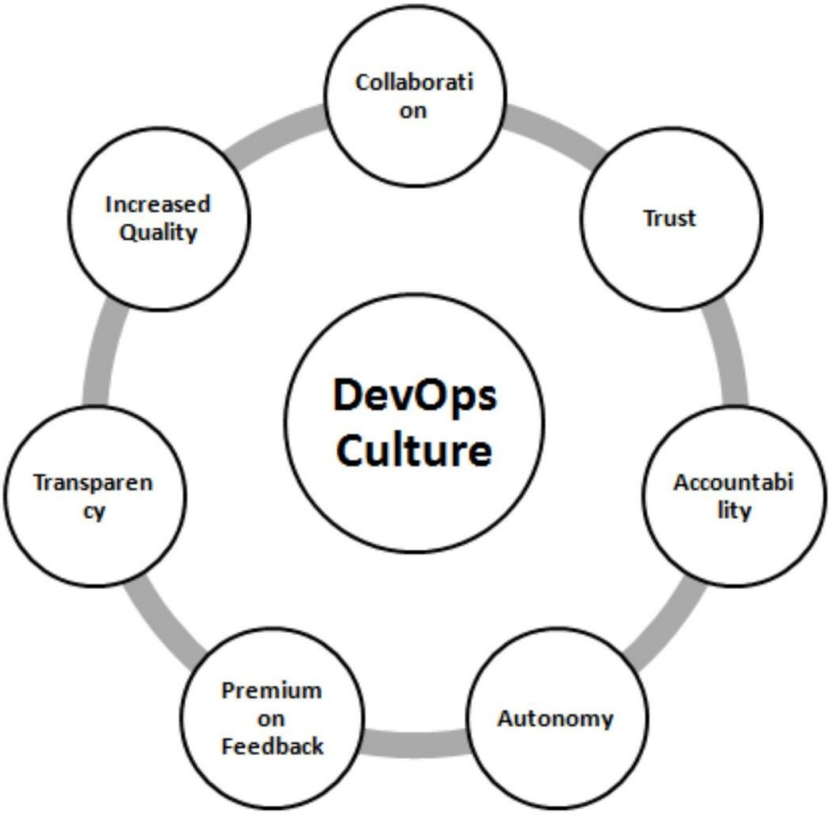
Agile practitioners will also tell you that you should never begin working in

iterations and adopt other practices or tools without starting the collaboration, as it will damage the process and the software. Many teams in various companies only adopt some methods of Agile but never worry about its principles. The results of such processes are suboptimal. You can use a tool in Agile, but if you are unaware of how to use it, you will undoubtedly damage the process.

In the end, it is not about fretting about which tools you should use for DevOps. If you were to plan an Agile process, do you think it would magically happen? No, you will need to use different tools that adhere to the Agile values. The same can be said about DevOps, a system that works only when its values are kept in mind.

It’s Not (Just) Culture

Most people insist that DevOps is a culture. They believe that one cannot apply that word to a principle or practice, but this is simply incorrect. Agile has helped many organizations and firms develop, but the development will reach a plateau since Agile stops at culture. This is because a lead practitioner in a team always decides what process the team should follow. They also refused to be more prescriptive. DevOps always contains rules or values for all the items in the list above. Since you will have enough time to experiment, you will be able to identify the best practices yourself, but it is always a good idea to fall back on some tried and tested policies before you start developing your own.



DevOps Culture

It's Not (Just) Developers and Operations

You must remember that it is not exclusionary. Some people claim that we don't care about network administrators, the security team and other smaller teams. The objective of DevOps is to create a system or product that can collaborate right from the start. This means that developers, operation teams, and business folk of various types need to work together to develop a strong process. These people include network administrators, security teams and every single other employee. There are different types of developer and business stakeholders, and just because they are not personally called for a discussion, it does not mean that they are not involved.

When Agile was first developed, people only cared about the collaboration between the business and the development team. DevOps always looks at issues and solutions around the collaboration between the development and operations teams. The result of using DevOps in your company is that everyone collaborates to develop the perfect product. From this perspective, DevOps is a

significant step that allows all disciplines within an organization to collaborate. Regardless of what role an employee or team play, if it does participate in the delivery of the software, it is a part of DevOps.

It's Not (Just) A Job Title

It does not make sense to take an existing operations team and rename it as a DevOps team. This does not help the situation at all, and it does not help to change the job title or role to DevOps Engineer. If you do not want to adopt any of the principles or values of DevOps, you will not reap any of the benefits since you will need to make changes to the overall system.

That does not mean that you cannot have a role that is called "DevOps Engineer." This title is often used to differentiate between the new style of thinking – where you first focus on automation, collaboration and the overall working of the software – from the old way of thinking where the person at the back-end does not necessarily care about what you do in your firm. Some people will find value in that, while others will not. If you are hiring a DevOps engineer, you should ensure that the applicant fits the role perfectly, and clearly know how they will work towards improving the process.

It's Not Everything

Sometimes, DevOps engineers get carried away and claim that this process regards every process everywhere. DevOps does plug into the overall structure of the organization, and it does use the principles of Agile and Lean thinking. There are opportunities to collaborate and build better and more efficient processes in an organization. It is always nice to see how you can use different tools in parallel; however, DevOps is not necessarily going to re-engineer all the business and its processes.

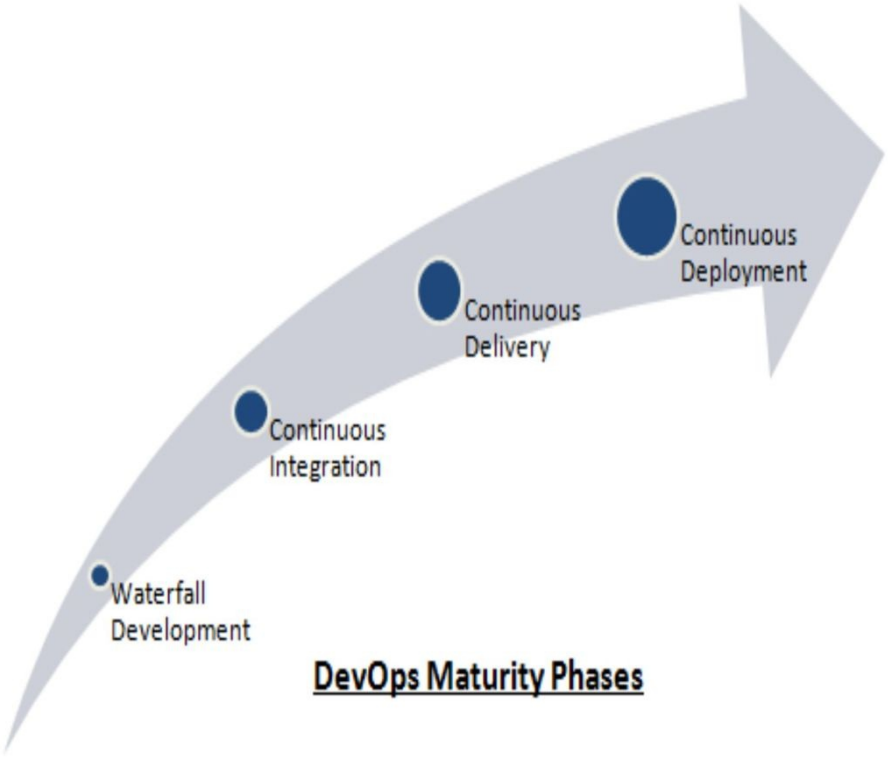
DevOps is a part of an overall agile and collaborative corporate culture, but it also looks at how different operations can plug into that. Some folks try too hard and turn the complete process into a watered-down version of Agile or Lean. This is a great vision, but when you move down and look at the granular sections of the process, you will see that you are just dealing with operational integration. There will still be a lot of problems that are not being addressed, and these issues revolve around the service maintenance, delivery of software and software development.

To increase the scope of projects, it is fine to use the different DevOps tools.

Most DevOps engineers, however, want to identify ways to improve their jobs rather than another person's. In Agile process management, there is Agile organization work and Agile Software Development. DevOps can be defined as an Agile Software Operations and Delivery process. This means that using DevOps, one should work on large organizational initiatives without losing sight of the primary values or objectives of the organization.

Getting Started With DevOps

You cannot follow a path to DevOps in your organization. All you need to do is look at different processes or tools that work for your organization. Most successful DevOps initiatives have originated from the collaboration of the development and operations teams, a bottom up and top down approach, from consultants and from inside the company, skunkworks pilots and widespread education. Therefore, it is always hard to follow steps to develop a process and implement it in your firm. It is safe to say that it always starts with the individuals in the organization. You will need to learn the values, principles, practices, and methods of DevOps and then try to spread that via an effective channel. This channel can be through the other members of the team, other employees in the company, and management. Alternatively, you can implement different strategies in the company by yourself and let the success speak for itself. People will always tell you how you can rise to success, but that advise is more wishful thinking and policy than reality. You should observe how some popular methods and tools rose in your organization, and try those channels. Always keep learning.



DevOps Maturity Phases

Chapter Three: An Introduction to Agile Management

To better understand DevOps, you should understand the different Agile techniques, as DevOps is an offshoot of it. Agile is an approach that adheres to twelve principles and the Agile Manifesto.

Pillars of Improvement

The Agile model is based on the following pillars, and DevOps and other Agile processes adhere to these principles.

Transparency

One feature of agile and DevOps is unfettered transparency. The information is passed from one member of the team to the next with ease. There are clear channels of communication. The organization will have a clear idea of what task each team is working on, and how many tasks are still in the product log. This is because the team works towards developing results right from the beginning of the cycle. Once the results or products hit the end users, feedback is sent to the team and improvements are made where necessary.

Transparency is not about viewing the results and quickly completing the work. Every member of the team should always look at the process through the same lens. The team should share a common framework and should collectively decide which task or requirement is a high priority. This will allow the organization and the team to know what the status of every requirement is.

Inspection

In the Agile framework, the tasks are broken down into smaller pieces of work. Every team is given a goal that it must achieve in a set period. This is called the sprint. When each requirement (also called a feature or user story) is completed, the team will test that feature to ensure that it works well. The team must also consider the customer’s requirements.

People who work closely with the project will always inspect the results. These people include the end user and the development teams. This will help to reduce the time lag, and it also means that any adjustments can be made to the feature immediately as the information is readily available.

Adaptation

If the team notices any inaccuracies or inefficiencies during the process, it should make changes to it immediately. This means that the team should learn to immediately adapt and make adjustments before it begins another task. In other words, teams should always understand which parts of the process are functioning well and which aren't. Through Agile and DevOps, the development and operations team have the right to inspect the work, thereby allowing the team to work collectively towards improving the process.

The Agile Manifesto

Agile is a framework that one can follow, and it should not be thought of as a methodology or process. You will need to make some choices. The biggest advantage of using Agile is that you can make some decisions that are best for the team using the feedback that comes from your end users.

The following are the values that form the Agile Manifesto, and every organization that wants to use Agile or DevOps must adhere to these values:

- Individuals and interactions between those individuals will take precedence over processes and tools.
- A working software model is more important to develop, rather than comprehensive documentation.
- Always collaborate with end users before a negotiation.
- Respond to change and adapt your process if necessary.

Although the Agile Manifesto and its principles were written by and for software, the values are valid for any project that your organization takes up. The GPS was initially designed for the military, but today, every individual in the world uses it to track someone or reach their destination. For more information on the history of the Agile manifesto and its founders, visit http://agilemanifesto.org.

Agile Principles

The founders or developers of Agile did not just stop at writing the values. They also defined some principles that teams must follow to adhere to those values. You can incorporate these values into your project and assess whether the framework you have developed adheres to these values.

- You should always satisfy the customer and deliver the software or product early and quickly. Alternatively, you can choose to deliver small sections of the product, called user stories or features to the customers.
- You should take customer feedback seriously and make changes to the product regardless of what stage the project is at. An agile process will allow you to make any change to the product if it improves customer satisfaction.
- Always deliver a working prototype of the software to the end user at regular intervals.
- Business or operations teams and development teams should always work together throughout the project.
- Ensure that the development team receives all the necessary support to ensure that the task is being done.
- Always have meetings where every team can convey information to other teams efficiently and effectively.
- Measure your progress by developing working prototypes of the product.
- Sponsors, users, and developers should always be on the same page throughout the project.
- Attention to detail and good design will improve the agility of the process.
- Following a simple process is important. This implies that you should identify ways to maximize the amount of work the team does not have to do.
- A self-organizing team can provide the best architecture and design.
- Teams should always reflect on what they should do to be more effective. They should then adjust their behavior accordingly.

The principles of Agile will never change, but the tools that a team uses to adhere to those principles can change. It is easier to implement some principles over others. For example, if you look at the second principle, the team can be open to new ideas and change. For such teams, it is easy to develop a framework and work on tasks. On the other hand, there will be teams who are more resistant to change.

Take a look at the sixth principle. Today, it is difficult for most teams to work face-to-face on a project due to the globalization of the workforce. You may have teams sitting in India, the United States of America and Russia. Instead of thinking that the principle cannot be applied to your team, you should look for a solution. Can you use Skype to stay in touch with the different members of the team? This is not what the sixth principle says, but if you want to improve processes, you should focus on how to deal with the principle today. This means that you should always learn to adapt to change.

Every team will face unique challenges, and you should never let a hiccup stop you from working on the project. Part of the fun is to work through issues and achieve the desired results. The same can be said about all principles. You can improve your team's quality and efficiency if you adhere to the principles listed above.

Platinum Principles

Experts suggest that a team or organization should abide by these principles when working on a project as they help to improve efficiency, as well as assist in the implementation process.

Avoiding Formality

Have you ever sat through a presentation and wondered how the presenter made such an amazing presentation? Do you wonder how much time that person spent to develop it? When you think that you are only preparing a presentation for a project, you are wasting your time. You can scribble whatever you want to say on a small sheet of paper and then get back to doing what you need to do. If you are concerned about a task or anything, you should speak to the concerned party and ask them to help you with it. You should always try to focus on your

objectives and not worry about how to prepare a presentation.

Atos Origin conducted a survey, and it concluded that an average employee spends at least forty percent of his or her productive hours on emails that do not add any value to the project or business. This means that for that employee, the workweek begins on Wednesday. People often confuse themselves with pageantry and progress. In an Agile system, you are encouraged to communicate directly and informally when you have a question. This will help you save time because you will work closely with every other member of the team. You have to always identify a simple way to achieve your goals.

Your projects will evolve and abide by the Agile values in no time. When people are more aware of the process, they work towards developing a robust or stable system.

Think and Act as a Team

The objective of Agile is to ensure that teams work well together. It is difficult to bring teams and individuals together since most organizations pit an individual against his or her peers. Individuals are encouraged to compete with their peers and are always questioning what they can do better to succeed in that environment. In Agile, the project will either thrive or die at the team level. When teams work well together and leverage the talent of an individual for the benefit of that team, they become hyper-productive. Aristotle once said, ‘The whole is greater than the sum of its parts,’ and this holds true for teams.

So, how do you develop this type of culture? The Agile framework only emphasizes on teams. Common goals, collective ownership, and physical space will improve the dynamics of a team. You should then add the following to your framework:

- Get rid of all work titles. No employee owns the team or some areas of development. One should establish themselves through his or her contribution and skills.
- Report only team metrics, and never individual or pairing metrics.
- Always pair team members to improve cross-functionality and improve the quality of work.

Visualize Instead of Writing

It is important to remember that people are visual. When they think pictorially, they will remember everything they have read about. Most children's books now have numerous images. As a kid, what did you like best about books? The text or the images? We are no different as adults. We will flip through a magazine to look at images, and if there is something that we like, we stop to read the article.

Diagrams, graphs, and pictures can relay information in an instant. If you write a huge report, people will stop reading because they know that there is no diagram to support any claim you are making. Twitter wanted to study the effectiveness of tweets without photos versus those with photos. It used SHIFT Media Manager to conduct this analysis, and it came up with some interesting results. A user would engage five times more frequently if a tweet included a photo. The rate of retweets for posts with photos were double those without. Also, they found that the cost per engagement of text-only tweets is double that of photo tweets.

You should always encourage your team to present information using visual aids, even if it means that they need to sketch a diagram on a whiteboard. If there is someone who does not understand the diagram, they can always ask you to explain. The same can be done with technology – you can either make models, charts or graphs to explain the technology.

Chapter Four: DevOps Principles and Concepts

DevOps Principles

Now that we have covered the Agile principles in the previous chapter, let's look at the most popular DevOps principles.

Customer Centric Action

It is essential to have short feedback loops where you directly communicate with end-users and real customers. You should also ensure that the activities that the team performs are only centered around these clients. If you want to meet your customer's requirements, you must act like a lean startup and always innovate your processes. If there is a strategy that is not working, you should have the guts to use a new one. You have to identify the products and services that delight your customers the most and invest in improving those products and services.

Always Create With The End In Mind

Every organization should now let go of the process-oriented and waterfall approach. In those approaches, an individual or a team works on a specific function or role without looking at the complete picture. Organizations should start acting like product companies and focus only on building working products. Employees should all share an engineering mindset, which will allow them to envision the final product.

End-To-End Responsibility

In traditional organizations, the development team will take care of developing IT solutions or products, and then hand those products over to the Operations team. The Operations team will then deploy those products to customers as well as maintain them. In DevOps, the teams are organized vertically, which means that every team in the organization is accountable. The products and services that are developed by these teams will always remain under their focus. These teams will also identify ways to support and improve performance, which will enhance the level of responsibility, thereby increasing the quality of products.

Cross-Functional Autonomous Teams

In most product organizations, the teams must be fully independent throughout

the process or lifecycle of the product. This means that they need to balance the skills in the team. This also means that the members of the team should be well versed and have enough information regarding what the other teams are doing. These teams will grow rapidly and develop products that will meet user requirements. It's possible that the feedback loop becomes smaller as the teams know what needs to be done to meet the requirements.

Continuous Improvement

Organizations must continuously adapt to ensure that they incorporate the changing circumstances including a change in customer needs, technology, and legislation. In DevOps, a strong focus is always maintained on the continuous improvement of processes to increase speed, optimize costs, improve delivery and minimize waste. The organization should also focus on improving the products or services that it develops. Therefore, it is important for teams to experiment to ensure that they learn from their failures and develop robust processes. A rule to live by is to fail more to learn more.

Automate Everything

Organizations should work toward continuously improving their processes. To do this, the organization must get rid of some waste. In the last few years, many companies have identified a way to eliminate waste from different processes to improve the products and services that it deploys. As an organization, you should always think of ways to automate the software development process and the infrastructure landscape by building cloud platforms that will allow you to view the infrastructure as a program or code.

Key Concepts and Metrics

Focus On Deployment Lead Time

Deployment lead time is a part of the value stream, and it starts when an engineer in the value stream wants to change a version of the software. The time taken for the engineer to push the change into the software, run that change in production and obtain feedback from the customer regarding the change is calculated as the deployment time.

Instead of processing large batches of work in short periods, the goal is to test

smaller batches of work. The goal is to test and operate the software while working on its design and development This will improve the quality of work and also enable the team to develop the software faster. That is why teams should break the project into smaller segments before they begin working on its development.

Process Time and Lead Time

In Lean, lead time is one of the measures or criteria used to measure the performance of the team. The other measure is the processing time. The lead time begins when the customer makes a request, and it ends when the request is met. The process time starts when the team begins to work on the customer's request. It does not consider the time when the work or process is in the job queue. Since the customer experiences lead time, we will focus on how to minimize it. Experts suggest that teams should look at the process time and identify ways to reduce it.

Chapter Five: Essential Skills for the DevOps Engineer

What does a DevOps engineer do?

DevOps is a joint process between software development and IT operations. Large companies are increasingly relying on the concept of DevOps to deliver high-quality products at an increased pace. DevOps engineers enable effective communication between these two entities to improve software development and delivery, enabling the organization to better and more effectively serve clients. They also facilitate continuous integration and development of output generated by back-end teams, ensuring a good pipeline for delivery of the code. Due to the integration of two separate processes, DevOps does not have a formal definition. Instead, DevOps engineers are either software developers who exhibit proclivity towards the operations aspect of a business, or operations professionals who are passionate about writing code and script.

(https://medium.com/@neonrocket/devops-is-a-culture-not-a-role-be1bed149b0)

Reflecting changes

It is difficult to track the changes in a traditional software development process. Changes in the code are possible but their reflection in a production environment is time-consuming, and this delays the release of the product. DevOps engineers can solve this by implementing a CI/CD pipelines to improve the speed of testing, development, and deployment.

Feedback and monitoring

Feedback loops developed for implementation, modification, and validation are

time-consuming. Now that the delivery pipeline is automated and there is constant feedback from the client, monitoring projects across the organization becomes easy.

Frequent and reliable releases

Updates, releases and version controls such as waterfall, spiral or iterative are critical points in development models. These decelerate the market availability of your application despite only needing a small bug fix. DevOps engineers can solve this problem efficiently and quickly, as they tend to work with the tools and strategies that bridge the gap between ‘development’ and ‘operation.’

What is a typical day in the life of a DevOps engineer?

The role of a DevOps engineer is centered on bridging product development and delivery while meeting the automation needs of other team members. Therefore, a typical day in a DO engineer’s life involves numerous meetings to ensure smooth inter-team collaboration and a focus on reducing overall manual effort.

How do I become a good DevOps engineer?

Successful DevOps engineers often possess most, if not all the skills listed below:

- The ability to collaborate effectively and skillfully
- The ability to facilitate seamless end-to-end communication
- Experience in development
- The ability to supervise infrastructure planning, testing, and development
- They ensure security and rapidity across the organization
- A clear understanding of DevOps tools, structure, processes and mindsets
- Anticipate needs, accumulate resources, and plan accordingly

- An eye to detect issues in automation, version control, and overall security and suggest necessary steps to solve them quickly
- Passion: At the end of the day, passion defines a good candidate, and it’s an asset that cannot be replaced.

What are the skills needed to become a good DevOps engineer?

Although there is no formal career track to become a DevOps engineer, the processes of development and operations can be used to define the skills required to become an accomplished DevOps engineer. Mainly, these skills are the following:

Technical Proficiency

DevOps is a highly technical field. If your background is in operations, begin sharpening your development skills with programming languages such as Python, Ruby, and Perl because you will be required to automate processes by writing code. Experience with scripting and administration platforms is also highly desirable to potential employers because server deployment is a significant bottleneck in the IT industry. Therefore, familiarize yourself with Linux and Ubuntu. On the other hand, if you are already a developer, learn how software delivery and deployment work. DevOps engineers must understand networking and storage because these are no longer silos. The need of the hour is to develop scalable solutions that function rapidly at any given time.

Automation Tools

As mentioned previously, the redundancy associated with many processes such as manual ones like assigning IP addresses and DNS reservation can be eliminated by scripts written in languages like Python and Ruby. Every DevOps role requires you to have some knowledge of deploying, operating and building software regardless of the type of stack used in the system. You must always be willing to apply the knowledge you have to use new tools whenever needed. Since most DevOps tools focus on automation, it is important that you familiarize yourself with different automation tools and techniques. This will

improve performance and remove the human component, thereby allowing the team to save some time. A DevOps engineer must be able to implement automation technologies and tools at any level.

Puppet, Vagrant

It is critical to maintain consistency in being available, as well as reliable and quick. Experience with these tools will help you repetitively and predictably manage software and system alterations.

Jenkins, Maven

Creating and deploying software is a key part of your job. With these tools, you will be able to ensure you have what you need to keep things going.

Git, Visual Studio Online and Perforce

These are examples of source control systems, and they allow for the collaboration of different teams on a specific project. These tools also make it easier for the organization to manage any updates and changes.

Nagios, Munin, Sensu

It is imperative that as a DevOps professional, you track and measure performance. You have to understand the principles and philosophy behind the many tools available at your disposal to help you implement the tool effectively.

Big picture thinking

Many DevOps engineers may be excellent software engineers, but sometimes, they cannot envision the larger picture and identify what makes sense to the client. As DevOps synergizes the needs and wants of multidisciplinary teams and applications, a successful DevOps engineer cannot lose sight of the big picture. This helps avoid the creation of contention points and constraints between projects.

Network awareness

A DevOps engineer should always be aware of the network.

Testing

Strong testing skills are essential to be a successful DevOps engineer, as automation is based on testing.

Soft Skills

DevOps engineers bring together different teams to work towards achieving a common goal. Working with people is key, and therefore, a successful DO engineer must possess strong communication and interpersonal skills. A great DevOps engineer is always curious to constantly learn new things, persistently generate questions and listen to answers to brainstorm for new ideas and welcome new challenges.

Security training

DevOps professionals do not need to be expert security professionals, but they will benefit tremendously from being trained in software security. This is because, at faster cycles, vulnerabilities in code can be introduced just as quickly. Therefore, DevOps engineers must be able to write secure code to safeguard applications against attack, as well as defend them against common cybersecurity vulnerabilities.

Customer-first mindset

A DevOps engineer does not necessarily have to maintain contact with external clients, but they will face different types of end users or customers. To ensure that the team maintains a strong relationship with the client, it is important to prioritize, empathize and be transparent regardless of the number of requests that come from the operations, development and product teams. A great DevOps engineer is always concerned about delivering a product with high quality and value to the customer. They also want to measure and understand how the end user perceives their work. By doing this, they develop the right instincts and know how to optimize the process to achieve success.

Collaboration

An organization that wants to implement DevOps must always put the team and the customer ahead of an individual in the team, and therefore strong DevOps professionals must be excellent team members and help widen bottlenecks by assisting co-workers when necessary. A DevOps engineer should also learn to multitask. DevOps engineers get requests from numerous clients at the same time and find themselves putting out fires at the same time. So, the ability to work on multiple tasks at the same time is crucial.

Chapter Six: Different Types Of Deployment

The four important types of deployment are:

Minimum In-Service deployment

In this type, the number of instances that stay in-service when you are building the application are mentioned. Therefore, this allows you to deploy as many targets as possible. This step is repeated until every server in the network has been updated.

Rolling application updates

Consider these as an extension of minimum in-service deployments. Instead of defining the number of containers that should remain online, we specify the maximum number of containers that need to be updated in parallel.

Blue/Green deployment

When following this method, an infrastructure is replicated for a short period. This replicated infrastructure now hosts the new application, while the old infrastructure continues running until testing completes and the new stack is incorporated. This was an expensive deployment method but not anymore, thanks to the Cloud. Once testing is finished, the application is switched over to the new version and the old stack is closed.

A/B testing

A/B deployments are similar to Blue/Green ones, except that a small amount of traffic is transferred to the new green environment. Switching environments in this method is more straightforward and precise way than Blue/Green deployment.

Lastly, deployment cycles may also vary based on the sensitivity and impact of the final product. For example, Amazon deploys new code around every 30 minutes on its website, but Southwest Airlines only deploys its patches twice a week as strict aviation laws are imposed to ensure rigid testing of aircraft systems.

Key Work Outputs

Overall, a DevOps engineer must ensure that the delivery pipeline is trustworthy and free of delays. On a day-to-day basis, other important contributions include:

- Guiding operations and development teams if there is an issue
- Reviewing, managing and monitoring technical operations
- Developing a continuous build environment to improve the speed of the development and deployment software
- Managing teams

Chapter Seven: DevOps Engineer versus Other Engineers

Difference between DevOps Engineer and Build and Release Engineer

DevOps Engineers and Build and Release Engineers have very similar roles, as they both need to identify ways to develop and deploy software. This chapter provides further information on what their responsibilities are and how different the roles are from one another.

Responsibilities of DevOps Engineers vs. Build and Release Engineers

Build and Release engineers and DevOps engineers must understand the technical requirements of a business or a project if they want to make it easier for them to build software. Some companies interchange these titles or bring a professional from each discipline onto the team. A DevOps engineer will create a pipeline that will be used to deliver stacks and tools to the development team. A build and release engineer will use various automation tools like Version One, Ansible or Jenkins to improve the process of building, testing and releasing the program to the end users.

DevOps Engineers

A DevOps engineer will work on Agile development teams and with other foundation architects and developers. They will design the build for the application and will suggest some simple, cost-effective methods to begin the process. These engineers work towards creating an automated development practice that will allow the team to develop the product faster and release it to the end user for feedback. Additionally, a DevOps engineer also improves the process of delivering the software to the end user. They do this by constantly updating the software since they will test every prototype during the development and delivery stages of the process. This will allow them to identify any operations issues and fix them. Some responsibilities include:

- Creating a style guide for applications and software.
- Maintaining the goals of the company to improve the performance of the product.
- Tracking any bottleneck issues using different deployment programs and updates.
- Understanding different tools like Puppet, Chef, and Bamboo.

Build and Release Engineers

A Build and Release Engineer will create and maintain the infrastructure which is called the source control system or the build system. This system is full of tools that will make it easier to build the software and deploy it to the end user. They will need to design the scripts and tools that the developers will use to build the software. These scripts can either be written in Java or Python. A part of their job is to identify a way to seamlessly integrate the many software updates by using continuous delivery pipelines. These pipelines will take the updates to the end user in a faster way. Some responsibilities include:

- Using Unix or Linux operating systems during the build process.
- Reviewing the code that is used for the build architecture.
- Creating add-ins or plugins that will improve the process.
- Looking for solutions to avoid automation issues and integrating those solutions into the system.

DevOps versus Site Reliability Engineering

There is a lot of confusion between the two titles, and this confusion has led to a debate about what these engineers do. In fact, many people believe that these roles are the same. There is a lot in common with these roles, especially when you consider the underlying objectives – automating, bridging the gap between the development and operations team, and scaling. However, there, is a significant difference between the two.

A dynamic organization that wants to scale at an aggressive pace will always want to lay the foundation or base for an IT department which can be both agile and nimble. For this, they will need to request the engineering department to create a foundation that can support their goals. The leads of these teams will

need to use automation tools to ensure that the infrastructure is widely available for multiple teams. There are two disciplines that one must consider in this instance – site reliability engineering (SRE), which is where an architect develops a fully automated infrastructure for IT, and DevOps, a combination of the Agile and Lean processes, which allows the developers to view the infrastructure as a piece of code.

Most often, IT infrastructure is built or constructed in small parts or segments as the company grows. The systems are all configured to ensure that they serve the day-to-day needs of the business. Also, the business will make changes to the infrastructure whenever necessary. A system administrator plays a major role in ensuring that the infrastructure is maintained regularly and system updates are always made to protect the infrastructure and maintain a productive environment. The administrators will spend hours, and sometimes days to ensure that everything is working correctly.

Sophisticated IT departments will use automated scripts right from the start to reduce the dependency of the infrastructure on the administration. The system architect or engineer can then include patches, management, and policy over the network using a centralized management system. This system can then be used to monitor the environment and detect any potential issues in the infrastructure before it causes any hindrance to the process.

An SRE is always focused on the role of a system engineer to maintain the infrastructure. This role applies to the production environment. DevOps is a practice that a business can use to automate or simplify a development team and reduce or remove their non-production environments.

There is a significant difference regarding the type of environment you are in and the focus on coding between DevOps and SRE. A DevOps engineer will always stick to the side of testing and creation since these departments are dynamic. These departments also embrace the Agile and Lean methods to help their teams run. This means that the processes must be automated.

A developer will leverage different automation tools like Chef or Puppet to assist or support the teams with this challenge. DevOps and SRE share common ground since the development engineer is always at the top of the pyramid. They will architect a culture and a system that will allow the organization to automate the delivery of tasks or infrastructure within the process.

In other words, new roles are being developed, namely the SRE and DevOps

engineers, to help the IT industry run efficiently. As the industry begins to grow, these practices will continue to evolve and newer practices will be developed. You must remember that it is operational efficiency that drives change. These forms of efficiency support the process of innovation and allow the IT department as a whole to run more fluidly.

Cloud Engineer versus DevOps Engineer

Both DevOps Engineers and Cloud Engineers use different computer languages like CSS and Java. The latter use these languages to design remote platforms where the data and files can be stored and accessed. The former creates the Agile team, which helps to facilitate the successful deployment of products and services by splitting the time that the developers spend on designing the software and deploying it to the customers. DevOps engineers can also assist with the development of cloud computing platforms, and they can work with downloadable software and online applications. Cloud engineers only design cloud-computing platforms for numerous companies. DevOps engineers, on the other hand, focus on improving the development and release cycle of software development by using the resources in the company effectively.

Cloud Engineers

A cloud engineer can create a remote network that a company can use to store, share and communicate data. Companies can also perform big data analysis, test and develop applications, and store back-ups of their data. Since the cloud architecture is massive, it can suit multiple users, and the cloud platform can handle large volumes of data. To become a cloud engineer, you must first understand the company's business plan and work towards creating a system that will fit their needs. If you want to learn more about the company, you should speak to the staff. This will help you determine the requirements, including what platform you want to use and why. As a cloud engineer, this will help you create a roadmap. You can then execute the design by developing a cloud infrastructure system. The job responsibilities include:

- Identifying creative cloud solutions
- Ensuring that the platform is compatible with Linux, iOS and Windows systems.
- Protecting the information stored over the cloud.

- Developing and monitoring the cloud.
- Updating drivers and firmware as needed.

DevOps Engineers

A DevOps engineer collaborates with the operations and development team to develop a rapid release and reliable pipeline for various software and updates. This involves the development of custom automation tools, the improvement of operations and the standardization of technology. As an Agile team member, the goal of the engineer should be to increase the speed of the processes that are involved in the creation and development of software. This will cover extensive technical responsibilities including automating the process for developers and tracking bugs or errors in the design. The engineer can also choose to document the procedures being used to build and maintain the deployment and configuration frameworks and create a guide regarding the best standards and practices. Some responsibilities include:

- Identifying new sections in the process to automate.
- Utilizing automated configuration tools like Puppet and Chef.
- Deploying and maintaining web-based applications.
- Monitoring different security issues.
- Measuring the performance of the team against any business outcomes.

Systems Engineer versus DevOps Engineer

Both systems engineers and DevOps engineers work only on office settings and primarily focus on technological advances. These engineers also attempt to bring different teams and departments together. DevOps engineers identify ways to connect the software development teams and deployment teams, while Systems engineers allow employees from different departments to share information and maintain transparency, thereby allowing them to work efficiently.

DevOps Engineers

A DevOps engineer works closely with software teams and uses different programming languages like Python and Java to create a cloud system or any other web-based application, like Netflix. These applications are already available to the user. The objective of a DevOps engineer is to develop software that will automatically update itself, which means that the user does not have to worry about updating the software. The engineer will need to identify a way to update the software remotely by using pipelines that will take the new code to the customer. It is always a good idea to create a product that is made up of

smaller modules or units. This will make it easier for the engineer to automate the process. Some key responsibilities include:

- Placing all the changes in the code in a single system, which will then be tested.
- Understanding how the software can grow when users save some information on it.
- Tracking the feedback loop and ensuring that feedback is taken into account before building the next user story or feature.
- Designing an environment for a new user, which is intuitive and automatic.

Systems Engineers

A Systems Engineer will design computer systems that bring software, hardware, and employees together. They will first consider the current system that is followed in the company, and identify the gaps in the system using data modeling. This method allows them to view the workflow. They will then work with the executives in the company and develop a contract that will outline the amendments that the new system should make and how it should improve the old system. This contract will also list the functions of the system. A system engineer will use this to estimate the cost of a project and develop visuals that the programmer can use. System Engineers also monitor the productivity and the reports from the new system to make modifications. The job responsibilities include:

- Gathering all the necessary input about the system by interviewing customers or end users.
- Suggesting or looking at alternative system elements to minimize cost.
- Building and developing prototypes.
- Supervising the integration to ensure that every part of the system works together.

Tools Engineer versus DevOps Engineer

A tool is a specialized software application that is designed to facilitate the work of artists, programmers, and designers. These tools are often found in the game

development world. It is the tool engineer’s responsibility to create and maintain software for the game studio. A game development studio uses many third-party systems like Houdini, Maya, and Unreal engines to bring environments and characters to life. Some studios still rely on in-house engineers to maintain and develop systems that are designed to meet the needs of a specific game.

Duties

A tools engineer is a professional who is hired to develop the foundation of a new software system. They will be involved in the design of the system during the initial stages of building the software. When a proprietary application is in place, the engineer is then brought on board. They are now responsible of helping the system adapt to the changing needs of the environment or the game studio by contributing new programs or codes. They must also identify ways to evolve the software, regardless of whether it is a porting pipeline 3-D art suite or a game engine.

A Tools Engineer is tasked with teaching every other programmer the fine points of the system. They should also work closely with character riggers, animators and other developers to fully understand the capabilities of the application. When a member of the designing staff insists on implementing new features in the game, the engineer must make those changes quickly and ensure that there are positive results. A similar position also exists in software companies in the animation, visual effects and game industries. The objective is to create a robust and functional software tool that will allow programmers, designers, and animators to work faster. This will ensure that the game reaches the hands of the customers quickly.

Skills & Education

It is always recommended that a Tools Engineer have a formal education in game design, software engineering or computer programming. They should also have a college degree in some related field, such as a certification in advanced mathematics.

A Tools Engineer should also know how to code in the C++ languages, and any other languages based on the game development studio. Employers always hire engineers or programmers who can write a program or code that is adaptable and portable. Your code should communicate with your fellow team members in the same way that you communicate with your superiors. It is also a good idea to

obtain a certification in game art or 3-D graphics.

Related Careers

If you want to become a DevOps engineer, you can also look at some other related careers like Scrum Product Owner. Both these professionals must be familiar with the requirements of the software. Regardless of what field you choose to enter, you can switch to other related careers if you do not enjoy your work.

Chapter Eight: Micro-Service and DevOps Engineers

Micro-Service Engineer

Microservices are services that can be deployed separately. These services perform specific business functions and only communicate over a web interface. A microservice is used to develop an application. A DevOps team will include smaller pieces of functionality in a microservice, and combine multiple microservices to build a large infrastructure. Therefore, you can view a microservice as a building block. There are many advantages that a microservice provides over monolith-type architecture. Microservices can remove SPOFs or single points of failure. They do this by ensuring that issues in one service do not affect any other service in the infrastructure. You can scale individual microservices to increase capacity. DevOps teams can improve or extend functionality by including new microservices that do not affect other parts of the application. Also, using microservices can boost team velocity.

DevOps Architect

DevOps Architects paint a bigger picture of the application and provide a foundation to build the product. They function in a manner analogous to an architect designing a house, in that they develop blueprints for an application, enabling a predictive framework to be formed. Overall, they bring in comprehensive practices that can establish a suitable DevOps channel across the entire organization.

DevOps in Banking

DevOps plays a big role in banking. An increasing number of banks are now shifting from on-premise server-based systems to cloud-based systems such as AWS, and therefore there is a constant requirement to hire DevOps engineers in this sector. Successful implementation of all DevOps targets is necessary because online banking portals are highly sensitive, meaning they need to be reliable, safe and secure. With customers trying to access their accounts and funds online, instant deployment is critical to avoid infiltration and exploitation of loopholes. For this reason, a DevOps engineer working with a banking institution must be able to work with cloud-based services and ensure smooth

and safe product deployment.

AWS Certified DevOps Engineer

AWS certified engineers are validated for their technical expertise in provisioning, operating and managing distributed application systems on the AWS platform. The concepts or techniques that one needs to master to obtain this certification are the following:

- Implementation and management of continuous delivery systems on the AWS platform
- Understanding and automating security controls and compliance validation
- Defining and deploying monitoring, metrics and logging systems on the AWS platform
- Implementing high availability, scalable systems
- Designing, managing and maintaining tools to automate operational processes

As AWS is now becoming the industry standard for DevOps practices, and DevOps engineers certified by AWS are highly sought after by other organizations looking to fill their DevOps positions.

Chapter Nine: DevOps Tools

This chapter lists some of the tools that a DevOps Engineer must be aware of.

Nagios and Icinga

Nagios is a tool that enables teams to monitor their infrastructure. There are many tools that one can use for this purpose, including Zabbix. Although many new tools have entered the industry, Nagios is highly effective since many members in the community create plugins for the tool. These plugins help DevOps engineers add new services or instances to Nagios which are not already present in the tool. Fortunately, this is not hard, and this tool can still be used to monitor infrastructure.

Icinga is an offshoot of Nagios. The creators of Icinga wanted to take Nagios to the next level with more modern features. This was done to enhance the experience of the modern user. There is still a debate in the community about which tool is better, but most companies and teams use Nagios since they are satisfied with its performance. There is a good chance that teams and organizations switch to Icinga in the future.

Monit

The simple watchdog tool Monit proves that simple tools are sometimes the most useful. The role of this tool is to ensure that any process on a machine is up and running without any errors. For instance, if there is a failure that occurs in Apache or Adobe, Monit will restart the process and make it work well with the software. One can set up Monit very easily and configure the settings to improve its efficiency. This tool is useful if you have a multi-service architecture and cater to various micro-services. If you use Monit, you must ensure that you monitor the restarts that it executes. This will help you identify problems and look for solutions instead of ignoring the failure. This can be done by monitoring the log files in the tool and ensuring that the tool alerts you whenever it restarts a process.

Elasticsearch, Logstash, Kibana or ELK

The ELK stack is one of the most common log analytics solutions developed in

the modern IT industry. This tool collects logs from different services, applications, tools, servers, networks and more, and stores it in an environment. This environment then moves that information into a single location, which can be accessed by a DevOps professional to analyze. You can use this information to troubleshoot problems, reduce the time taken to solve problems and monitor services. You can also use this tool for auditing and security, which means that you can change settings and monitor the changes made in security groups. You can act on any of these activities when you receive an alert on the issue. Some companies use ELK for business intelligence and use it to monitor the users and their behavior. You can either purchase ELK as a service or set it up on your system.

Consul.io

Consul is a great tool that you can use to discover services and configure applications that are built from several micro-services. This is an open-source tool that uses the latest technology to provide internal DNS names for a variety of services. It acts as a broker and helps you sign and register different names thereby allowing you to access servers instead of machines. Let’s assume that you have a cluster of machines. You can use Consul to register these machines as one unit and access that cluster with ease. There is a lot more that you can do with this tool, and you can experiment with it to make it work better.

Jenkins

Regardless of whether you are aspiring to be a DevOps Engineer or are still a developer, you have definitely heard about Jenkins. This is not the fanciest or fastest tool, but it is easy to use it since it has a great ecosystem of add-ons and plugins. This tool is built in a way that makes it easy for a user to customize it to their purpose. You can customize the tool to create and code Docker containers, run numerous tests and improve the production or staging process. This tool is great to use, but there are some issues regarding the performance and scaling of the tool, which is not unusual. You can also explore CircleCI and Travis, which are tools that require little or no maintenance.

Docker

There are many articles available on the internet that talk about how Docker has transformed the IT environment. The tool is life changing and great, but people

sometimes experience challenges with it. Most businesses use Docket in their production services, as it eases control issues and configuration management and also allows scaling by moving containers from one place to another.

Some businesses have developed a SaaS solution using a twelve-layer data processing pipeline. When you use Docker and Jenkins together, you can run the full pipeline across several, if not all layers, on a single system. There are some complications with using Docker since it takes time to build even the smallest container. As a DevOps engineer, you will also want to ensure that your developers can work rapidly and are satisfied. This can be a challenge since it is difficult to manage storage, networking, and security for a container.

Experts believe that there will be further updates made to the tool, and they welcome new management solutions using the updated version of the tool.

Ansible

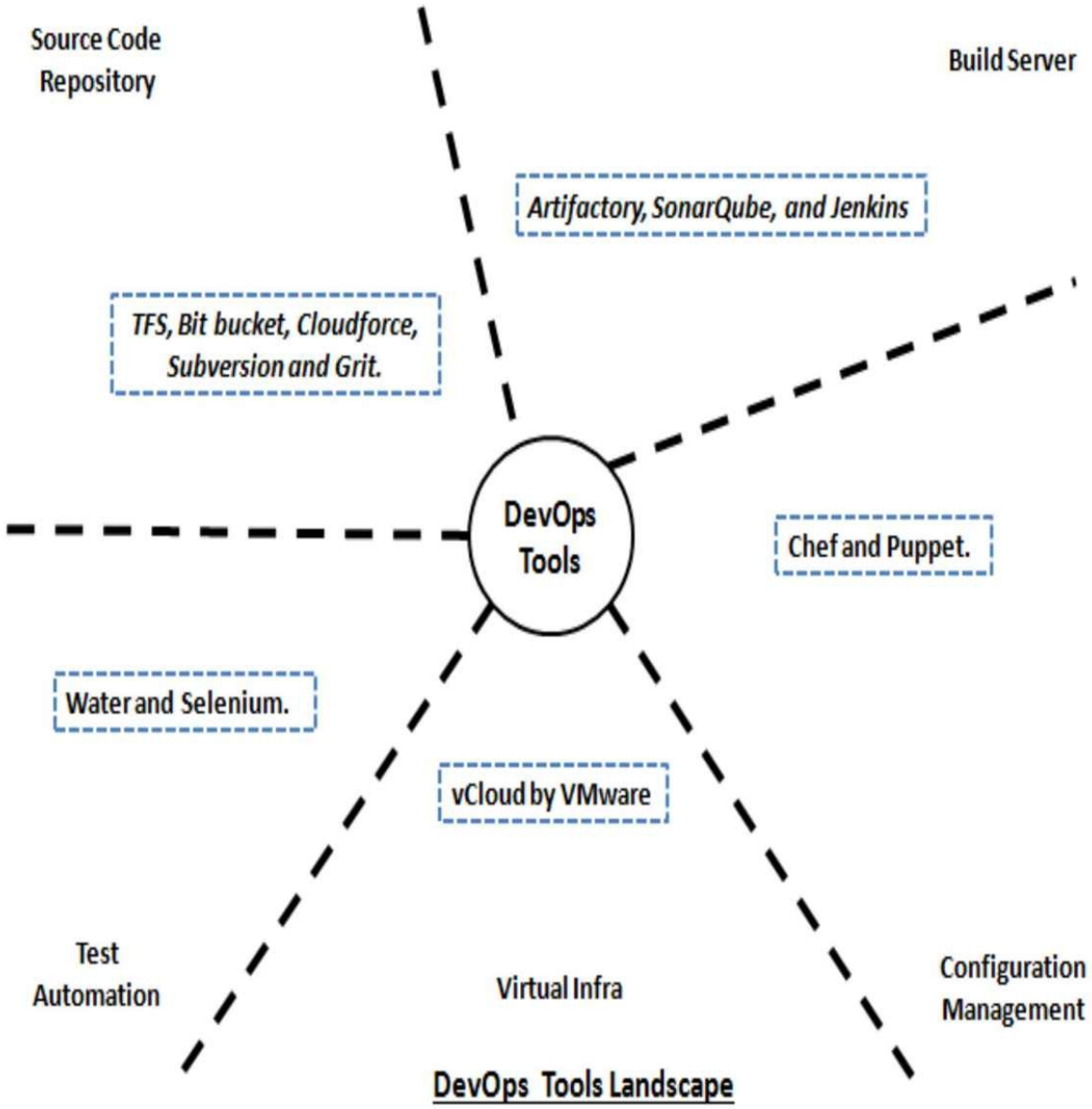
Once again, simplicity is key. Ansible, like Puppet and Chef, is a configuration management tool. Experts find that Puppet and Chef are more complex to use, and suggest that one learn how to use Ansible. Puppet and Chef have a wider range of features, but it would be best to start with Ansible since it is the easiest configuration tool to learn. There are trade-offs to using Ansible where you can kill an application and build a new one using Docker containers. Through Docker, you can spin new machines, which reduces the need to upgrade the machine to different cloud instances. Ansible is often used to configure the deployment process. This tool is used to push and re-configure newly deployed machines, and the ecosystem allows users to write custom applications.

Collectd and Collectl

Collectd and Collectl are little tools that derive and store statistics about the systems on which they run. When compared to other tools, they are flexible. Using them, a user can measure a variety of system metrics. These tools are unlike other logs in the sense that they can measure multiple parameters in parallel. Some systems use these tools to measure performance parameters, and these values are then shipped to the ELK-as-a-service platform. Some organizations wrap a Collectd or Collectl agent in a container and then push it to all the servers using Ansible. The agent will collect information every few minutes or seconds and ship that information to the ELK. This will allow the business to send alerts and run reports whenever necessary.

Git (GitHub)

The Linux Community needed a Source Control Management (SCM) software through which one could support distributed systems, and this is why GitHub was created. GitHub is the most common source management tool, and it is used by many companies. It was run internally for a short period before the developers realized that they could make it a hub. GitHub has amazing features like pull requests and forking. Apart from this, GitHub can connect to Jenkins and help with the integration of tools and the deployment of the product or service. This is a tool that most people may have heard about, but it should be a tool you know well since it adds great value to you.



DevOps Tools Landscape

Chapter Ten: The Career Path

A DevOps practitioner is one of the highest paid IT professionals in today's market. The demand for these practitioners is increasing rapidly since many organizations have started to use DevOps practices to improve their processes. A report published by Puppetlabs: State of DevOps stated that organizations that use the DevOps approach deploy their code at least 30 times more frequently than any other company. In addition to this, their code only fails less than fifty percent of the time. Over the last two years, the listings for DevOps jobs on different portals like Indeed.com and LinkedIn.com have increased by fifty percent.

DevOps Engineer Salary

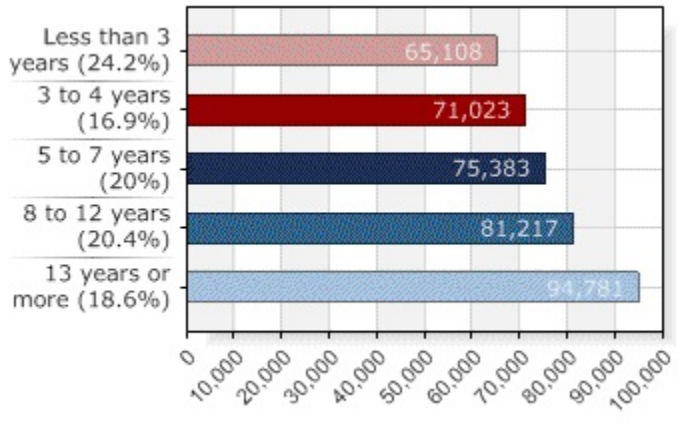
The graph below was sourced from Indeed.com, and it shows that the DevOps jobs began to increase from the year 2015 onwards. Towards the end of the same year, the number of jobs available in the market went through the roof.

(https://www.edureka.co/blog/devops-engineer-career-path-your-guide-to-bagging-top-devops-jobs)

The salaries for DevOps jobs also increased during the year 2015, and it continues to increase today.

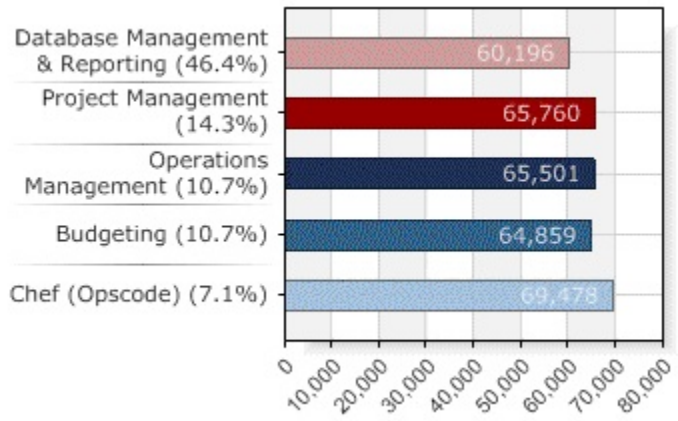
(https://www.edureka.co/blog/devops-engineer-career-path-your-guide-to-bagging-top-devops-jobs)

According to the website PayScale.com, the monthly salaries for DevOps professionals starts from $94,000, depending on experience.



(https://www.edureka.co/blog/devops-engineer-career-path-your-guide-to-bagging-top-devops-jobs)

Unlike jobs in every other technological field, the DevOps field is now witnessing an impressive trend. All jobs that fall under the DevOps bracket share similar roles and opportunities. For example, if a project management role had 65,000 openings and budgeting roles had 64,000 openings, jobs around the DevOps tool (Jenkins or Chef) were higher at 69,000.



(https://www.edureka.co/blog/devops-engineer-career-path-your-guide-to-bagging-top-devops-jobs)

DevOps Roles and Responsibilities

Many new roles are now emerging around the DevOps philosophy. These include:

- DevOps Architect
- Automation Engineer
- Software Tester
- Security Engineer
- Integration Specialist
- Release Manager

DevOps began as an ideology or framework that different organizations could use to bring different schools of thought in an organization together. It is now a popular career path that has its own perks, and the opportunities are increasing by the day. It is truly the right time for you to start exploring the path of a DevOps engineer. You can begin your training now and take advantage of the many opportunities that are available to you.

How Do You Start?

You may have many questions. How do you prepare for an interview? How do you know where to start? How do you land a job?

The Career Credential Challenge

Unfortunately, there are very few formal training programs in DevOps, but it is easy to find a training school and obtain a certification for many IT careers. If you want to code or write programs, you should go to college and obtain a degree in Computer Science. Alternatively, you can teach yourself how to code and apply your skills to a startup or a small company that does not worry about credentials.

If you want to work on system administration, you can master in Computer Science and obtain additional certifications in Linux or Unix. You can also become a database administrator by taking up courses available on the internet. When it comes to some form of training in DevOps, there are not many colleges or schools that offer training and certification.

No institution offers higher learning in DevOps or even courses in it. There are only a handful of certificate programs and tests that are available on the internet, but these are from organizations that do not have any DevOps career opportunities.

Alternatively, you can become a certified DevOps professional for platforms like Amazon Web Services, but cannot become a DevOps professional as a whole.

Start Developing Skills

In the earlier chapters, we covered some of the important skills that a DevOps professional should have, so you should focus on how to develop those skills.

College Courses

It is always a good idea to start with a degree in Electrical Engineering, Computer Science, or a similar field. It may not provide all the information that you need to begin as a DevOps engineer after graduation, but it still gives you a strong technical foundation. This is true if you choose to take a course that covers numerous areas including philosophy or history, instead of focusing only on operating systems or programming.

DevOps MOOCs

You can find many open online courses (MOOCs) on numerous websites. EdX, Udacity and Coursera offer DevOps courses. Most employers may not be impressed if you complete a MOOC since a college degree will take precedence over that certificate, but it can still help you stand out. You will also get training in DevOps, which is oriented towards real-world applications instead of focusing only on theory like in college.

Public Demonstration of Skills

If you have written a tool that will solve a software management task, you should ensure that you publish it on GitHub. If you have thoughts on how to coordinate software delivery across different systems, you should write a blog and share that information. You should always be ready to demonstrate your achievements and skills.

The article you write or the code you develop does not necessarily have to go viral for you to advance in your work. When you make your work publicly available, you can use it as a passive indicator of your expertise and also show

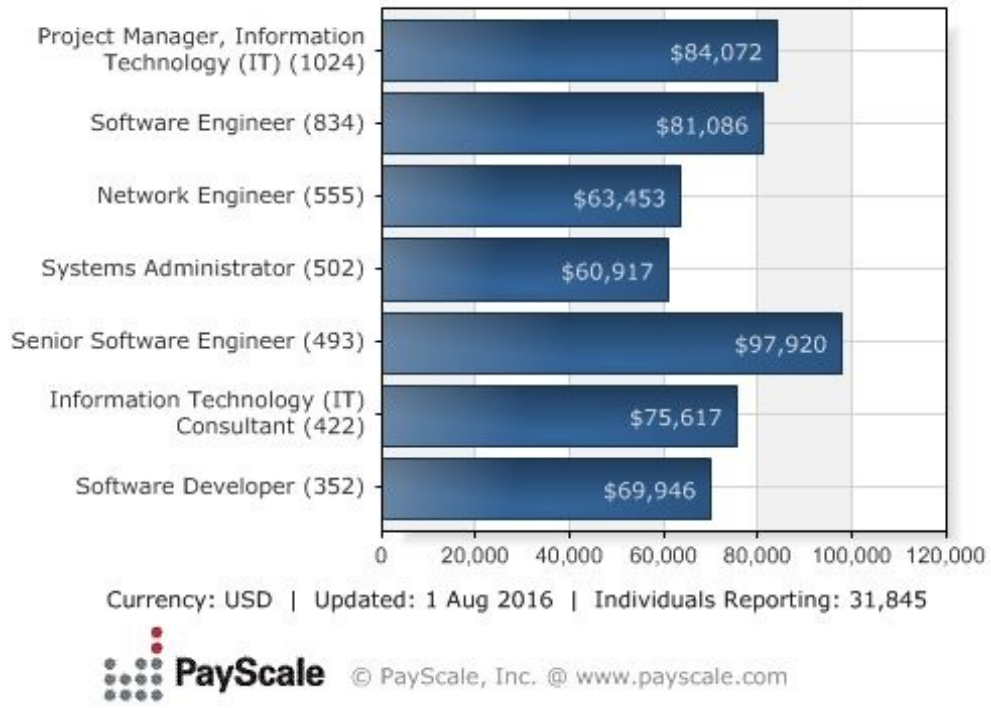
the industry that you are committed to contributing to the community. This is always a good thing.

Following the DevOps Conversation

You have to ensure that you are always up to date with the conversation taking place in the industry. People no longer talk about DevOps the way that they used to. This process not only involves the IT operations team or the development team but also includes the security engineers, quality assurance team and more. You should always strive to learn more about DevOps. It is harder to start a DevOps career than it is to land a technical job, but you can do it if you find the right educational resources and use those resources in the right way. Ensure that you immerse and involve yourself in the DevOps community.

Chapter Eleven: How To Market Yourself As A DevOps Engineer?

In today's world, the DevOps career path is a satisfying career option and one of the most profitable options in the IT industry. According to a survey conducted by Incapsula, the median of salaries for DevOps careers is between $104,000 and $129,230. The salary is dependent on the size of the team. When you compare the salaries on Payscale.com, you will see that there is a stark difference between the salaries of DevOps professionals and non-DevOps IT professionals.



(https://techbeacon.com/5-ways-make-yourself-marketable-devops-engineer)

How to Become an Indispensable DevOps Engineer

As mentioned in the earlier chapter, it is important to build your profile to land a

DevOps job. So, what should you do to build on the skill sets and the experience that you need? How do you get your personal brand out into the market to land a job? This chapter lists some of the best ways to make yourself an indispensable DevOps engineer. This input is based on sayings by experts in the field.

Start Cross-training now

Regardless of whether you are an operations manager or a developer, it is never too late to acquire skills that are outside your current role and comfort zone. This is only if you want to become a DevOps professional. A developer has an edge over the Operations professional since they have experience in coding and programming. Also, they understand the needs of a developer. It is always great if a DevOps professional was previously a developer rather than just an IT professional.

A DevOps engineer with a developer background will have more in-depth knowledge and understanding about the tools that the developers use. Therefore, they can leverage that knowledge to improve processes. The key is to always look for a different way to apply a concept from a previous role to a new position.

You have to use your experience and skills as a software engineer or developer to build tools, and you should not focus only on the development of software. For some examples, you can look at the many open-source projects that Netflix has written. If you want to become a DevOps Engineer, you should try to work on operational tasks. If the team does not work on any such tasks, you should sit with the operations team and see what work they do during deployments.

Some managers refer hiring developers as DevOps engineers; however, some managers will still consider you for the position of a DevOps engineer if you have experience as a system administrator. All you need to do is work on your scripting and programming skills. In the past, many system administrators were called upon to automate tasks like configuration, provisioning, and employment since it was difficult for them to develop scripting skills. This means that they must always put in the time to gain experience on Puppet or Chef scripting, develop a familiarity with Shell and Linux environments and gain some knowledge and experience in programming languages. They can learn languages like Python, Perl, Ruby, and Bash.

You should always document your work in the form of code and spend sufficient time understanding automation frameworks that will help you automate deployments and provisioning.

Build Skills in the Job You Have For the Job You Want

Many companies offer the position of a plum DevOps engineer, and they only select applicants who have already worked with DevOps. An IT professional who does not have a DevOps title will find it easier to land these roles if they can include DevOps-minded tasks and responsibilities to their existing roles. They should do this regardless of whether they change their job title or not. Take John Behling as an example – Field Nation who was recently hired as the Senior DevOps engineer for Field Nation. Before this, he worked as a System Administrator since 2008.

John mentioned that he always spent some time to automate tasks to make his life easier. He also mentioned that things began to change in the year 2014 when he began the role of a System Administrator. In this role, he was expected to develop a server that several development teams could use. It was then that he began to call himself a DevOps Engineer, as he was being used to complete tasks for the new-school system administration work.

Automate Processes

Rick Tai, Head of Engineering at Reflektive, mentioned that he always hires people who can complete cumbersome processes within a few minutes or hours. This means that he will only hire a DevOps engineer who can use automation tools to manage deployments and servers. Behling agreed that it is important for an aspiring DevOps engineer to master different automation practices and tools, as this will improve their marketability and increase their salary. Behling says that one should always study automation, and believes that one should never be stuck with tedious manual work since they will not be happy about where they are. If you want to automate tasks, you should have a deep understanding of the task. So, when you spend time on automation, you will gain new insights into the technology that you are supporting.

Involve Yourself with the Community

As mentioned in the previous chapter, you should always involve yourself with

the DevOps community, as you can only market yourself if you build on your skills and identify a way to display those skills. You can only do this if you are involved with the DevOps community. There is not just one way to go about this, since a process depends on every individual team member's skills and interests. You can either speak at industry events or write blogs or articles. Alternatively, you can stay in touch with your colleagues through different online platforms. Behling says that you should also try to work on an open-source project since that will have a larger impact on the community.

When you work on an open-source project that is related to DevOps, you will be giving yourself a big opportunity since you can show the world what you know and what you can do with your skills. Presenting at meetups and talking in small gatherings is important. Also, you can participate in public Slack channels and answer questions. This will help you improve your brand and visibility.

Eric Sigler, Director of Site Reliability at PagerDuty believes that DevOps engineers do not always have to be involved in the community. He believes that people who do not see community involvement as mandatory gain more from it. He believes that one should not approach the community only to develop a career. People should always use the community to connect with other professionals and gain exposure to new ideas. Since every person in the community brings their enthusiasm, they can provide a unique support system.

Demonstrate Empathy and Curiosity

Sigler believes that empathy and curiosity are two soft-skill traits that a DevOps Engineer should demonstrate. If a person can demonstrate these qualities, an organization will trust that the person will care deeply about the company, their role in the company and the DevOps processes.

It is easier to demonstrate curiosity since you can always develop or build a wide skill set, and also learn how to ask the right questions when you are working on a project. This will help you change for the better. It is a little difficult to show empathy since this means that you need to identify a way to build muscle memory in the ear, as listening is a big part of developing empathy. Another way to develop empathy is to put yourself in situations where you will need to communicate with others. It is crucial to translate what you do to your peers – presenting, explaining and teaching. This is because you may find a way to improve the process or logic while talking to someone else about it. You must ensure that what you are doing is not something that only you understand and

that whatever you do is accessible to every member of the team.

Chapter Twelve: Interview Questions

This chapter lists some questions that you may be asked in an interview.

Question: **Explain or define DevOps.**

Answer: You should ensure that you cover some or all of the points mentioned below:

- DevOps improves the collaboration and communication between the operations and business development teams. Through DevOps, you can unify different teams within an organization.
- DevOps is not just about the tools. It is mainly about involving different people and the process.
- The Lean and Agile processes inspire DevOps
- Through DevOps, you can automate the development, operation and release processes.
- Through DevOps, you can increase the speed of delivering applications to end-users.
- DevOps focuses on the improvement of processes through continuous feedback loops.

Question: **What are some prerequisites that are necessary for DevOps Implementation?**

Answer: The prerequisites include:

- The need for change must be communicated across the organization
- The senior management should commit to the values of DevOps
- Versions of the deployed software
- Automated testing
- Automated tools to maintain compliance
- Automated deployment

Question: **What are the best practices of DevOps implementation?**

Answer: The implementation of DevOps will vary for different organizations. In today's market, organizations and businesses want to deliver their products faster. This answer focuses on how you can use DevOps to improve communication between the organization, including the operations team, and the customers.

- The DevOps implementation should always align with the business goals. This means that the organization must agree that there is a need for change.
- The implementation should encourage collaboration and communication between the operations and development teams.
- It is important to automate processes, and during the SDLC stages, one should automate processes whenever possible. This will facilitate the implementation of DevOps.
- Tools that are used in DevOps should integrate and work well together. You should always look for tools that come from one vendor or look at open source tools that integrate with your organization's values.
- You should always include the CI and CD practices or tools in your implementation. Therefore, you should identify ways to continuously integrate code with testing, and continuously deliver the product to customers.
- The operations team must ensure that the applications work well at all levels. They may need to work with the development team if necessary to build or construct tools that will help to build or incorporate the right monitoring methods into various applications.
- Encourage feedback from your customers to ensure that there is continuous improvement. This will ensure that the process improves continuously and that quality of the software improves.

Question: **What are the components of DevOps?**

Answer: The following are the components of DevOps:

- Continuous monitoring
- Continuous integration

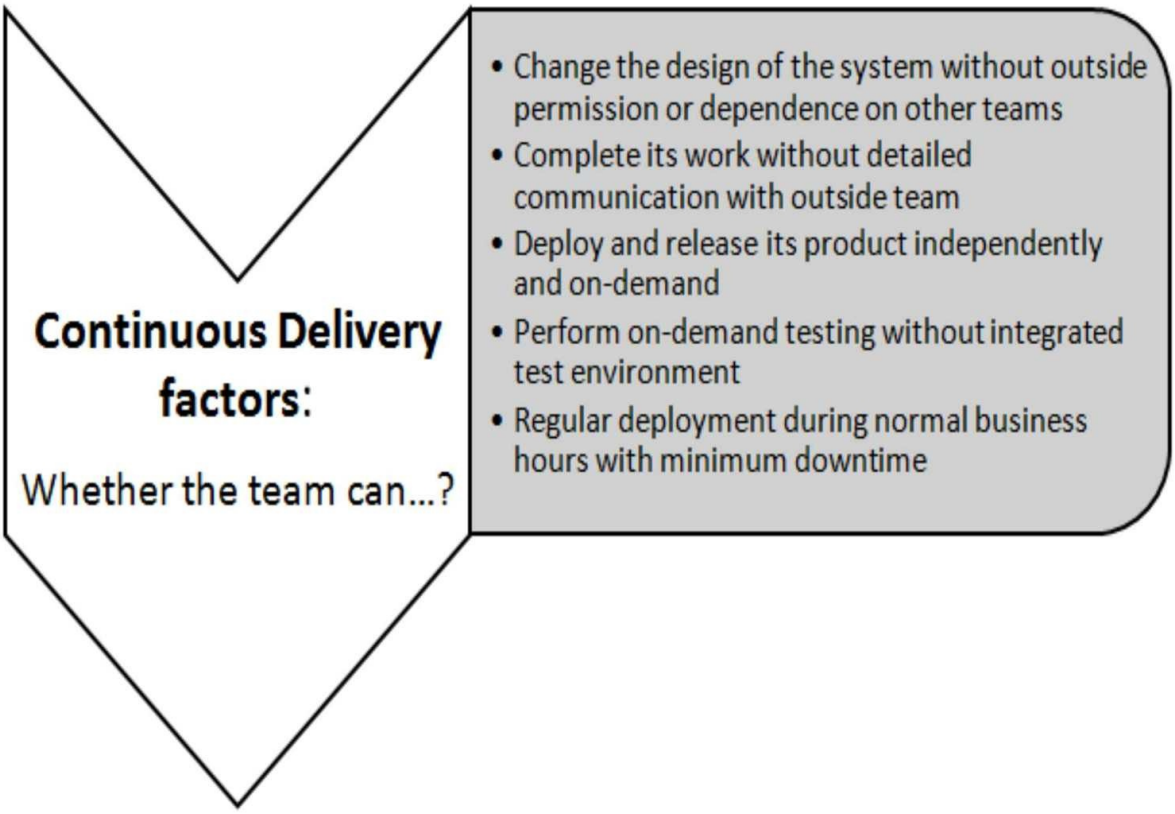
- Continuous delivery
- Continuous testing

Question: **Define Continuous Integration.**

Answer: Continuous integration is an important part of the Agile process. A developer will work on a user story or a feature during a sprint and make changes to the developed version continuously. When the code is committed, the work of the developers is integrated with the final product. The team will then perform the build regularly based on a schedule or check-in. Therefore; continuous integration is a practice that forces developers to integrate the changes they make to the product with other changes to obtain feedback from customers.

Question: Define **Continuous Delivery.**

Answer: Continuous delivery ensures that the features that the developers build reach the customers very quickly. This is an extension of continuous integration. It is during this process that the changes go through testing and quality assurance tests. When there are no further changes to make, the product is moved to the production system where it will be deployed.


Continuous Delivery factors:
Whether the team can...?
• Change the design of the system without outside permission or dependence on other teams
• Complete its work without detailed communication with outside team
• Deploy and release its product independently and on-demand
• Perform on-demand testing without integrated test environment
• Regular deployment during normal business hours with minimum downtime

Question: **Define Continuous Testing.**

Answer: As mentioned earlier, teams do their best to make changes to their products and deliver them to their customers very quickly. This can only happen when the product goes through sufficient automation and unit tests. Therefore, the team should identify ways to validate the code or script and ensure that the code is in line with the vision.

Question: **Define Continuous Monitoring.**

Answer: When the application is fully developed and deployed, the team must monitor how it performs in the market. It is important to monitor the products since it will help the team identify defects at an early stage.

Question: **How should you approach a project if you need to implement DevOps?**

Answer: You can use the following approach to implement DevOps for a project, but this approach will vary depending on the organization or the project.

- Stage 1: Before you implement DevOps, you should assess the existing process and see how it is implemented for at least three weeks. You can use at least five applications to see how the process works, identify any improvements and also develop a roadmap that you can use for the implementation of DevOps. This assessment should be performed by a senior DevOps architect.
- Stage 2: You can then perform a pilot POC that you can use to show the end-to-end working of the DevOps implementation. When the customers accept this implementation, you can begin the actual implementation of DevOps. You should also hand the project over and roll out a plan for the various projects involved. You can also include training sessions whenever necessary to understand the various tools and the process.
- Stage 3: You should now implement the DevOps model, and run all processes through that implementation. This means that you should follow the values of DevOps.

Question: **Can one apply the DevOps process to a traditional process or waterfall process? What is the significance of the Agile movement in DevOps implementation?**

Answer: In the waterfall process, the first step is to gather requirements. Once they are gathered, the team will develop or design the system, implement it, and test it. The final step is to deploy the final product to the customers or end users. The problem with this process is that there is a significant gap between the building and deployment of the developed product, which makes it very hard to get feedback from the end users.

The solution to this is to use Agile processes to improve the functions of the operations and development team. You must abide by the principles of the Agile process to ensure that the DevOps implementation works. This is because DevOps is an offshoot of the Agile process.

DevOps is focused on releasing the software promptly, which allows the team to

obtain feedback quickly.

Question: **State the difference between continuous delivery and continuous deployment**.

Answer: If you look at an Agile sprint, there are many user stories or features that the team will develop, test and deploy to the end user. Based on the priorities and needs of customers, not all user stories or features are deployed. Therefore, in continuous delivery, it is important to ensure that the code is always ready to be deployed.

When it comes to continuous deployment, the changes developed go through various stages of testing before the user story or feature moves to the production environment.

Question: How much experience do you have working on DevOps projects?

Answer: You should explain to the interviewer what your tasks were and what your role was as a DevOps Engineer. You should also explain what type of environment you worked in, and the projects that you worked on. If you have automated process before, you should also provide information about it and talk about how you supported other teams. This will show the interviewer that you took up enough responsibility and also identified ways to automate processes in the organization. You should also talk about some of the Agile processes and tools that you may have used in end-to-end automation. If you have experience working with cloud computing platforms like AWS, you should let the interviewer know about that too.

Question: **What are the most important DevOps tools that are used in the industry?**

Answer: You can list the following tools:

- Jira
- Bamboo
- GIT/SVN
- Artifactory/Nexus
- Jenkins
- Bitbucket

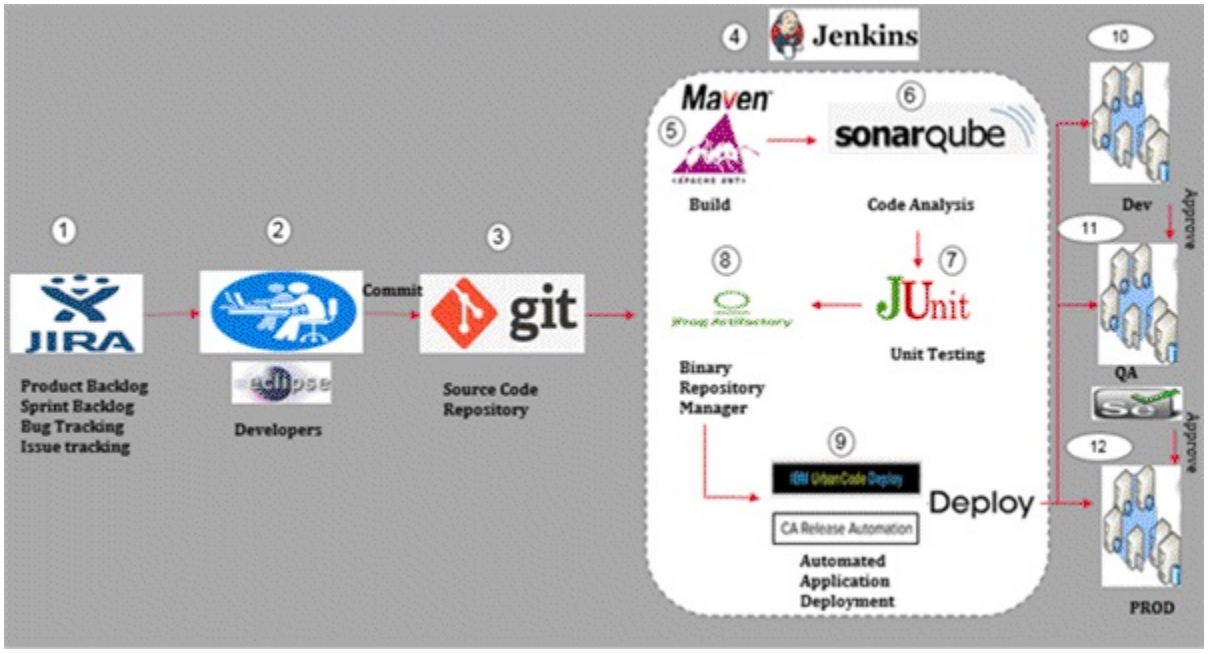
- Docker
- SonarQube
- IBM Urbancode Deploy / CA-RA
- Chef / Puppet /Ansible
- Nagios / Splunk

Question: **Explain the uses of some or all the tools that you have mentioned above, and how you can use them in the DevOps model.**

Answer: If you know how or why the different tools are used, you should let the interviewer know immediately.

- **Jira**: This tool is used for issue management and project planning.
- **GitHub**: This tool is used for continuous integration and version control
- **Jenkins**: This is an open source tool that is used for continuous integration. It can also be used in continuous delivery.
- **SonarQube**: This tool is used to analyze code.
- **JFrog Artifactory**: This tool is used as a binary repository manager.
- **Chef/Ansible/Puppet**: These tools are used for continuous delivery, application deployment, and configuration management.
- **IBM Urbancode Deploy/ CA RA**: This tool is used for continuous delivery.
- Nagios/Spunk: This tool is used for continuous monitoring.

You can now shed some light on a sample workflow of the DevOps implementation.



(https://www.edureka.co/blog/interview-questions/top-devops-interview-questions-2016/)

- In most agile processes, the tasks, defects and user stories are stored in JIRA. These processes are then assigned to developers and product owners.
- The developers will select the tasks that are assigned to them and work on development. The source code or primary version of the product is stored in the Git. The developers will make necessary changes to the code in Git, and it is then shared with the other developers through GitHub.
- Jenkins is a continuous integration tool which will pull the code out on every run or check-in. This is dependent on a schedule that the team builds using tools like ANT and Maven.
- Once the J2EE WAR files are produced, they are controlled and stored in a binary repository manager like Nexus or Artifactory.
- Unit testing will use Code Analysis and Junit along with SonarQube. This process is also automated.
- When the above process is fully complete, the continuous delivery section of the process is performed. This is performed on different environments based on different tools like Deploy / CA RA or IBM

UrbanCode. Continuous Testing is used in appropriate test environments.

- Continuous monitoring is an ongoing activity that takes place in the production environment.

Question: **Define configuration management in terms of infrastructure. List the popular tools used.**

Answer: Configuration management uses different practices and tools to automate the delivery and infrastructure operations. This process is about maintaining the server to ensure that it is always ready to be used. The tools used in this space are Chef, Ansible and Puppet.

Question: **Which scripting tools should you use in DevOps?**

Answer: Ruby and Python

Question: **What are the roles involved in DevOps?**

Answer: We have covered this in the first few chapters of the book. There are two roles involved in DevOps:

- DevOps Architect: The person responsible for the complete DevOps process.
- DevOps Engineer: This person should know Agile, Lean, SCM, CI/CD and should know how to set up automation tools. He should also possess database management and infrastructure automation skills. A person who has skills in coding and programming, and knows how to deploy projects or work as a system administrator can become a DevOps engineer.

Question: **Have you ever used the cloud to develop or implement DevOps? If yes, which platform or software have you used?**

Answer: You should familiarize yourself with the following computing platforms:

- Microsoft Azure
- Amazon Web Services (AWS)
- Google Cloud

Question: **List some of the metrics that aided in the success of DevOps.**

Answer: Some of the examples that you can mention are listed below:

- The most important factor that one should consider is the speed of delivery. This means that the developers should not take too long to complete an item and push it into the production environment.
- The next aspect to consider would be the deployment of the user story or feature, and how long it would take if the process was automated.
- It is also important to track the number of defects or bugs identified in the environment with respect to the production environment. This is extremely important since it will consider features that need to be released sooner rather than later. Since DevOps uses Agile principles, it helps to reduce defects in the production level.
- Deployments do not fail, but it is important to keep track of this. The DevOps team should identify a way to roll back or move to a previous stable version of the software.
- In a DevOps implementation, it is important to test the unit and ensure that the functional aspects of the code are tested. Depending on the changes made to the code, the DevOps team should look at what tests it will need to conduct and to what extent. It is important to ensure that the test is robust enough to withstand any changes made to the code.
- It is extremely important to identify a way to measure the average and actual time the development team takes to recover the software if there is any failure in the environment. This is defined as the mean time to recover (MTTR), and it should be very short. This means that the teams should identify a way to monitor the process.
- The performance of the application is a key metric that must be monitored, especially when a deployment is made.
- An important factor for the success of the DevOps tool is the number of bugs that the customers report. If there are too many bugs in the system, the customers will question the quality of the application.

Question: **What do you expect from a career as a DevOps professional or engineer?**

Answer: The answer here can vary depending on what you want to achieve as a

DevOps professional, but you should cover the following points:

- Improve processes to enable the collaboration between the operations and development teams.
- To help different teams understand each other’s visions.
- To be a part of the end-to-end delivery of the product.

Conclusion

Thank you for purchasing the book.

In recent times, many tools have been developed to improve different processes. DevOps is one of them, but most people consider it a managerial tactic rather than a tool. Through the implementation of DevOps, a business can ensure that the development and operations teams collaborate to build the best software products. This book covers all the information you need to understand what DevOps is and how you can use it to improve your business.

Hello!

How was it?

If you liked the book kindly leave a review!

Any further suggestions: Kindly reach out at valueadd2life@gmail.com

See you soon,

Stephen Fleming

Sources

https://www.techrepublic.com/article/10-critical-skills-that-every-devops-engineer-needs-for-success/

https://study.com/articles/devops_engineer_vs_build_and_release_engineer.html

https://www.linkedin.com/pulse/devops-vs-site-reliability-engineering-sean-washington/

https://study.com/articles/cloud_engineer_vs_devops_engineer.html

https://www.quora.com/What-is-a-tools-engineer-How-is-it-different-than-a-DevOps-engineer-or-network-engineer

https://study.com/articles/devops_engineer_vs_systems_engineer.html

https://dzone.com/devops-tutorials-tools-news

https://theagileadmin.com/what-is-devops/

https://dzone.com/articles/devops-devops-principles

https://www.devopsagileskills.org/dasa-devops-principles/

https://dzone.com/articles/an-introduction-to-devops-principles

https://techbeacon.com/5-ways-make-yourself-marketable-devops-engineer

https://logz.io/blog/become-devops-engineer/

https://devopsagenda.techtarget.com/opinion/Looking-for-a-DevOps-career-path-Heres-how-to-get-started

https://www.edureka.co/blog/devops-engineer-career-path-your-guide-to-bagging-top-devops-jobs

https://www.edureka.co/blog/interview-questions/top-devops-interview-questions-2016/

https://www.softwaretestinghelp.com/devops-interview-questions/

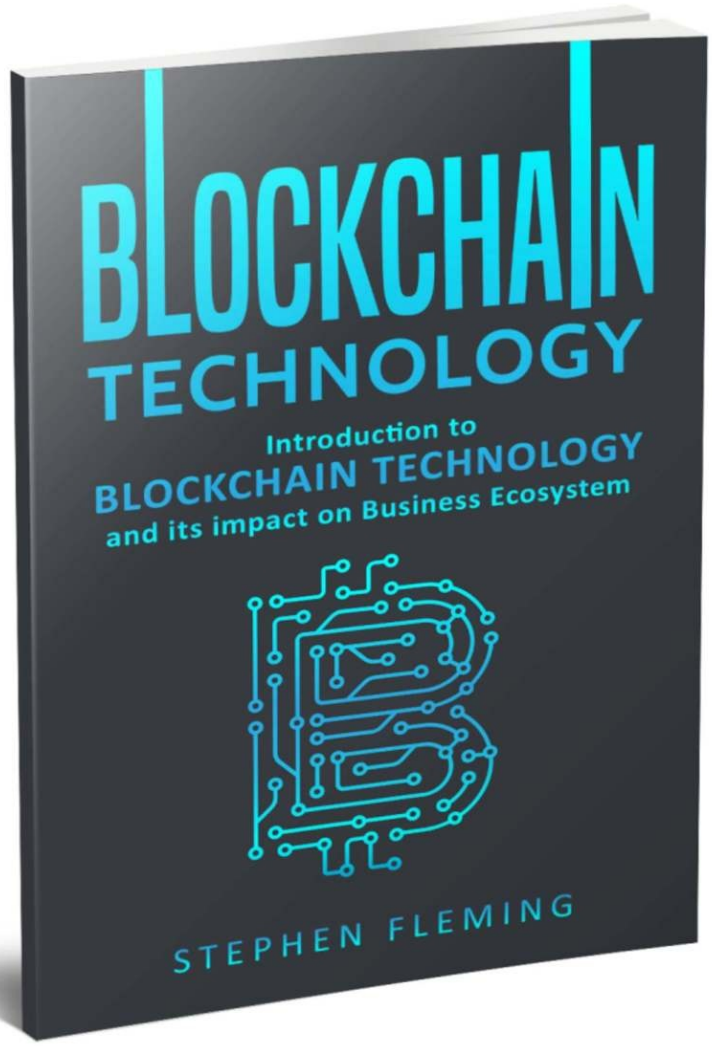
https://www.quora.com/How-does-a-DevOps-engineer-use-Jenkins

https://devops.com/9-open-source-devops-tools-love/

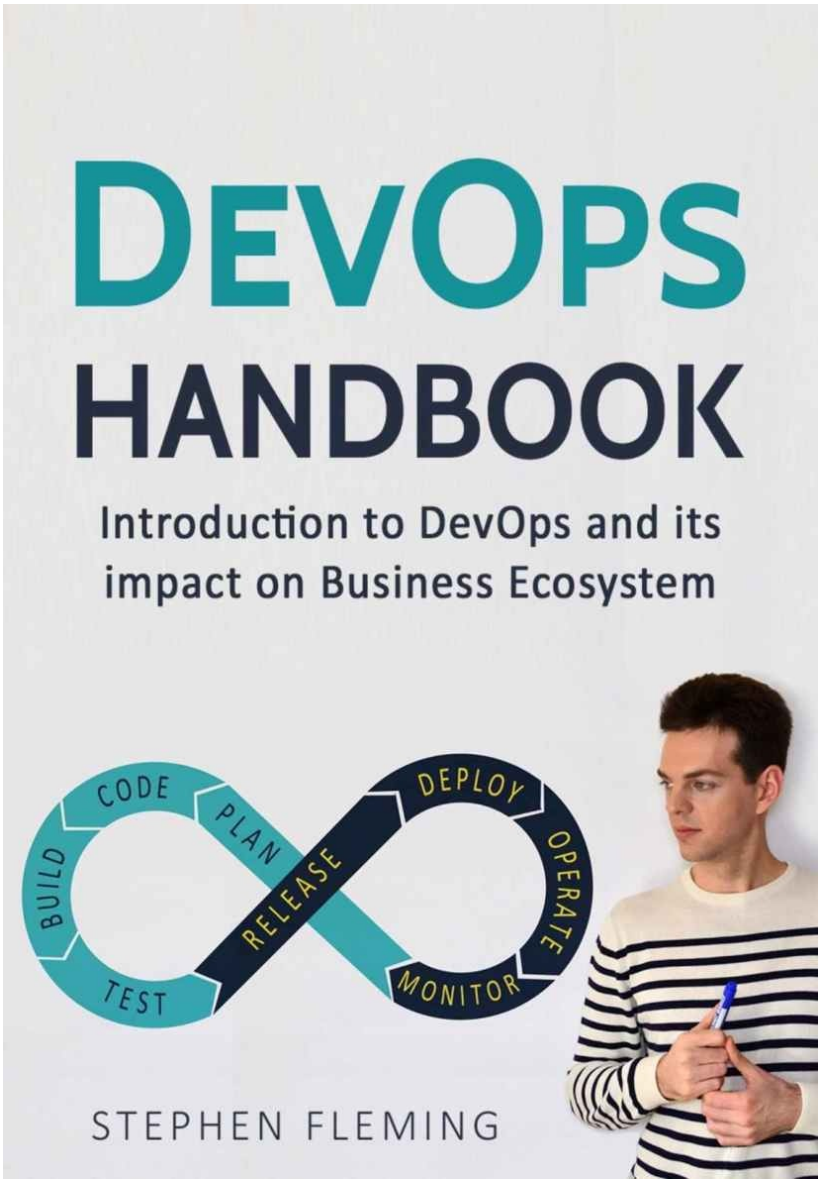
My Other Books available across the platforms in e-book, paperback and audible versions:

*You can check all my Books at my author page: Click Here

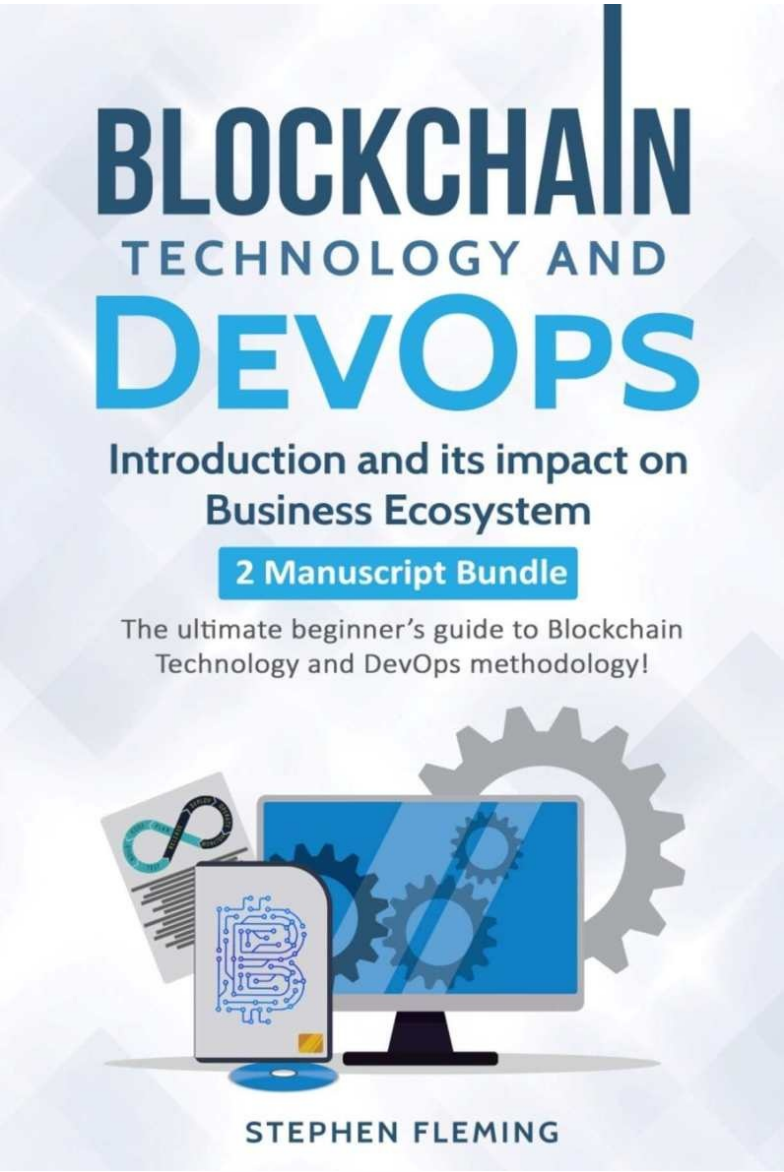
Blockchain Technology: Introduction to Blockchain Technology and its impact on Business Ecosystem



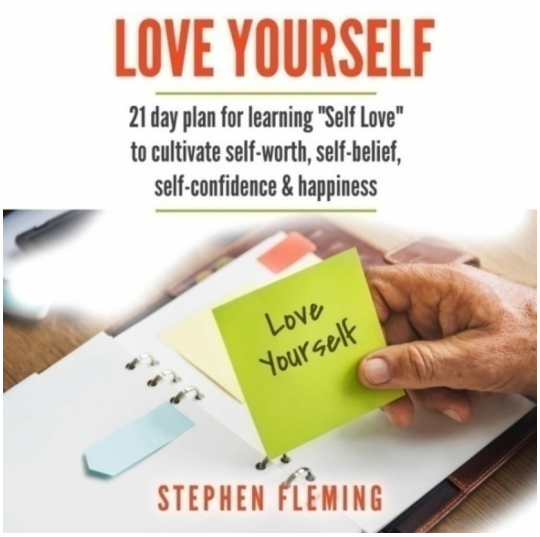
DevOps Handbook: Introduction to DevOps and its Impact on Business Ecosystem



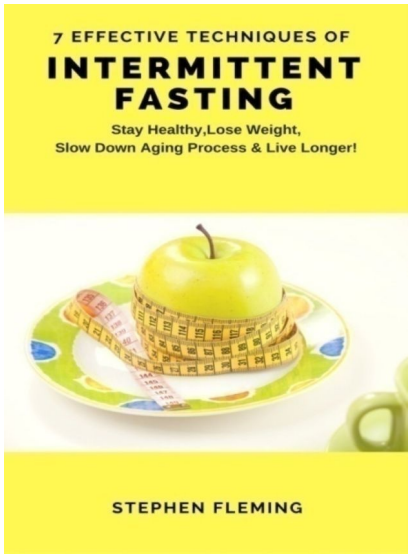
Blockchain Technology and DevOps : Introduction and Impact on Business Ecosystem



Love yourself: 21 day plan for learning "Self-Love" to cultivate self-worth, self-belief, self-confidence & happiness

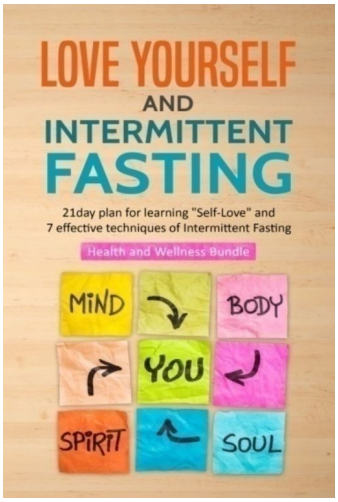


Intermittent Fasting: 7 effective techniques of Intermittent Fasting

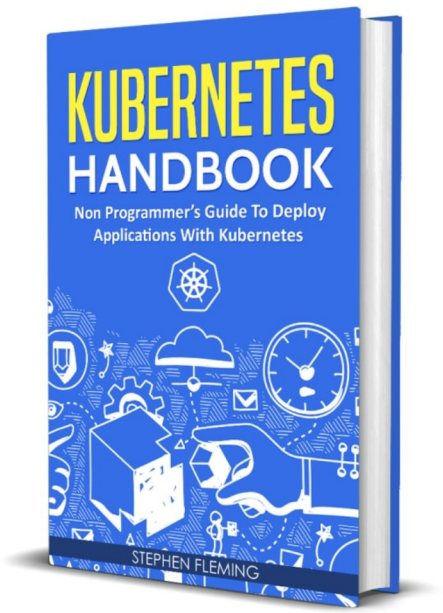


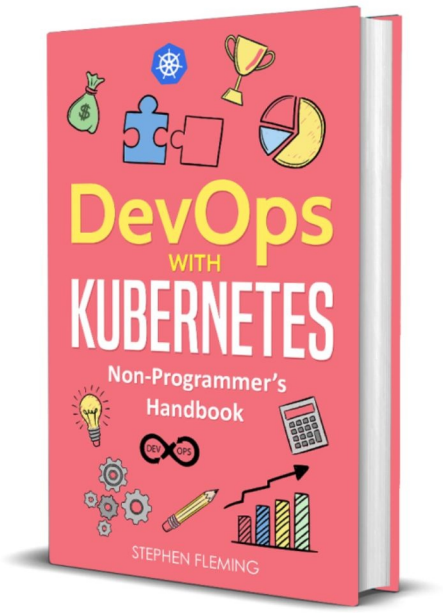
Love yourself and intermittent Fasting (Mind and

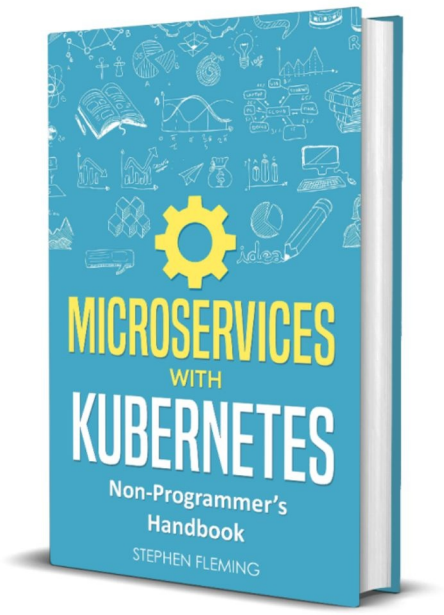
Body Bundle Book)

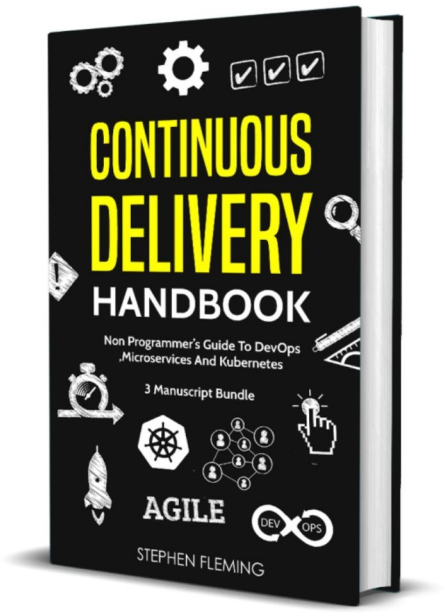


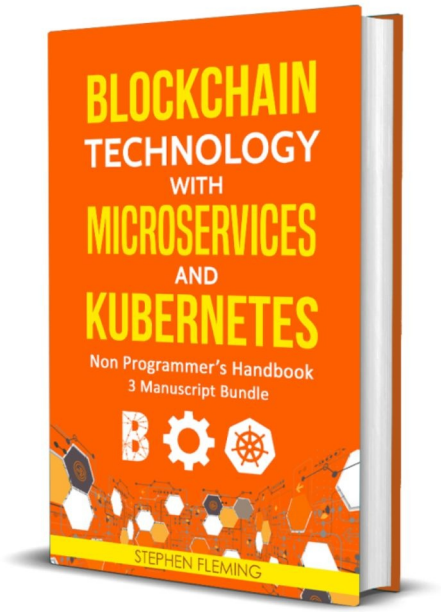
New Releases

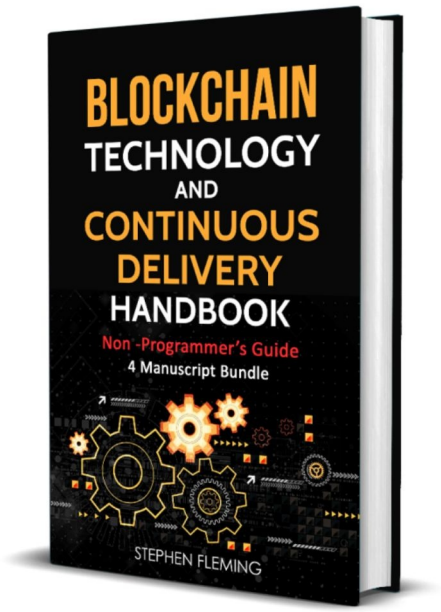


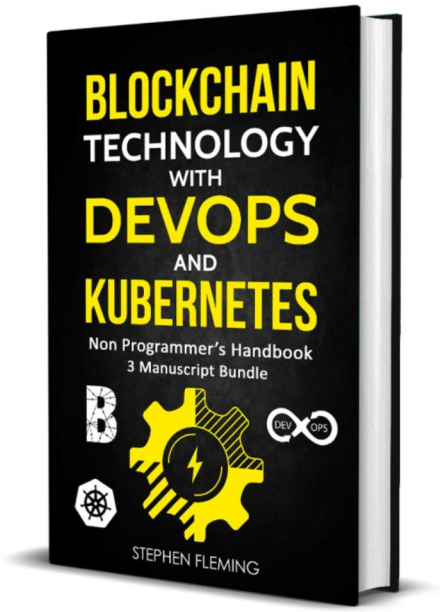


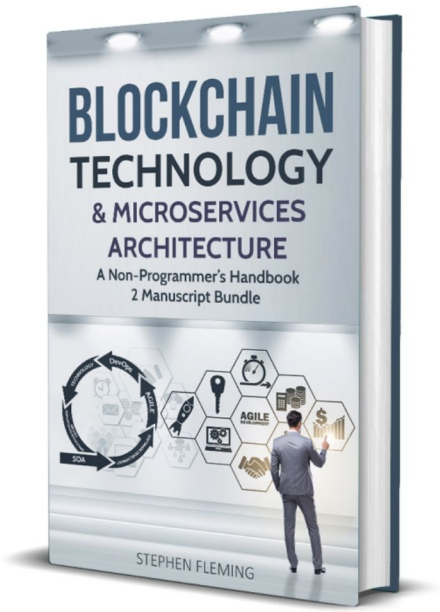

MICROSERVICES
WITH
KUBERNETES
Non-Programmer's
Handbook
STEPHEN FLEMING

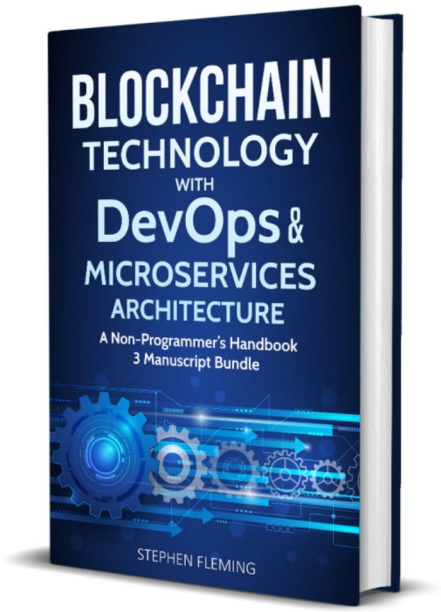

CONTINUOUS
DELIVERY
HANDBOOK
Non Programmer's Guide To DevOps
,Microservices And Kubernetes
3 Manuscript Bundle
AGILE
DEV
OPS
STEPHEN FLEMING

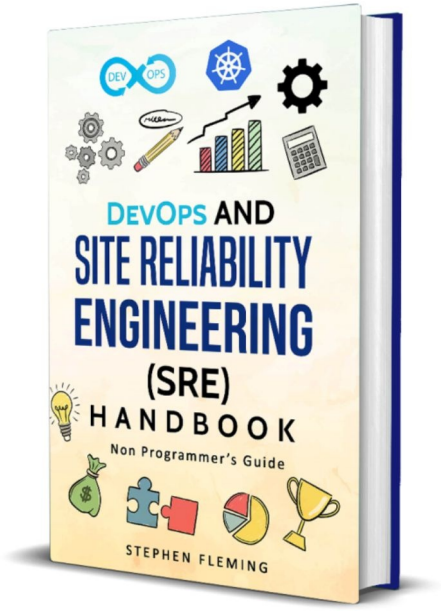

BLOCKCHAIN
TECHNOLOGY
WITH
MICROSERVICES
AND
KUBERNETES
Non Programmer's Handbook
3 Manuscript Bundle
STEPHEN FLEMING

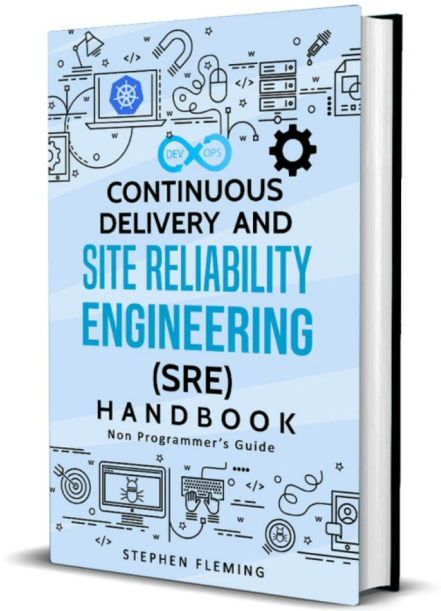

BLOCKCHAIN
TECHNOLOGY
AND
CONTINUOUS
DELIVERY
HANDBOOK
Non -Programmer's Guide
4 Manuscript Bundle
STEPHEN FLEMING


BLOCKCHAIN
TECHNOLOGY
WITH
DEVOPS
AND
KUBERNETES
Non Programmer's Handbook
3 Manuscript Bundle
DEV
OPS
STEPHEN FLEMING


BLOCKCHAIN
TECHNOLOGY
& MICROSERVICES
ARCHITECTURE
A Non-Programmer's Handbook
2 Manuscript Bundle
AGILE
SOA
STEPHEN FLEMING


BLOCKCHAIN
TECHNOLOGY
WITH
DevOps &
MICROSERVICES
ARCHITECTURE
A Non-Programmer's Handbook
3 Manuscript Bundle
STEPHEN FLEMING


DEVOPS AND
SITE RELIABILITY
ENGINEERING
(SRE)
HANDBOOK
Non Programmer's Guide
STEPHEN FLEMING



** If you prefer audible versions of these books, I have few free coupons, mail me at valueadd2life@gmail.com. If available, I would mail you the same.

*You can check all my Books at my author page: Click Here

If you liked the book, kindly leave a Review!